Deep Neural Networks
and
Tabular Data:
Inference, Generation, and Explainability

Deep Neural Networks and Tabular Data: Inference, Generation, and Explainability

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät

der Eberhard Karls Universität Tübingen

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

(Dr. rer. nat.)

vorgelegt von

M.Sc. Vadim Borisov
aus Woronesch, Russland

Tübingen
2023

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	04.05.2023
Dekan:	Prof. Dr. Thilo Stehle
1. Berichterstatter:	Prof. Dr. Gjergji Kasneci
2. Berichterstatter:	Prof. Dr. Andreas Zell

"It takes something more than intelligence to act intelligently."
– Fyodor Mikhailovich Dostoevsky,
Crime and Punishment, 1866

To the power that brings us further.

Abstract

Over the last decade, deep neural networks have enabled remarkable technological advancements, potentially transforming a wide range of aspects of our lives in the future. It is becoming increasingly common for deep-learning models to be used in a variety of situations in the modern life, ranging from search and recommendations to financial and healthcare solutions, and the number of applications utilizing deep neural networks is still on the rise.

However, a lot of recent research efforts in deep learning have focused primarily on neural networks and domains in which they excel. This includes computer vision, audio processing, and natural language processing. It is a general tendency for data in these areas to be homogeneous, whereas heterogeneous tabular datasets have received relatively scant attention despite the fact that they are extremely prevalent. In fact, more than half of the datasets on the Google dataset platform are structured and can be represented in a tabular form.

The first aim of this study is to provide a thoughtful and comprehensive analysis of deep neural networks' application to modeling and generating tabular data. Apart from that, an open-source performance benchmark on tabular data is presented, where we thoroughly compare over twenty machine and deep learning models on heterogeneous tabular datasets.

The second contribution relates to synthetic tabular data generation. Inspired by their success in other homogeneous data modalities, deep generative models such as variational autoencoders and generative adversarial networks are also commonly applied for tabular data generation. However, the use of Transformer-based large language models (which are also generative) for tabular data generation have been received scant research attention. Our contribution to this literature consists of the development of a novel method for generating tabular data based on this family of autoregressive generative models that, on multiple challenging benchmarks, outperformed the current state-of-the-art methods for tabular data generation.

Another crucial aspect for a deep-learning data system is that it needs to be reliable and trustworthy to gain broader acceptance in practice, especially in life-critical fields. One of the possible ways to bring trust into a data-driven system is to use explainable machine-learning methods. In spite of this, the current explanation methods often fail to provide robust explanations due to their high sensitivity to the hyperparameter selection or even changes of the random seed. Furthermore, most of these methods are based on feature-wise importance, ignoring the crucial relationship between variables in a sample. The third aim of this work is to address both of these issues by offering more robust and stable explanations, as well as taking into account the relationships between variables using a graph structure.

In summary, this thesis made a significant contribution that touched many areas related to deep neural networks and heterogeneous tabular data as well as the usage of explainable machine learning methods.

Kurzfassung

In den letzten zehn Jahren haben tiefe neuronale Netze bemerkenswerte technologische Fortschritte ermöglicht. Sie haben das Potenzial, in Zukunft eine Vielzahl von Aspekten unseres Lebens zu verändern. Deep-Learning-Modelle werden immer häufiger in verschiedenen Situationen des modernen Lebens eingesetzt, von der Suche und Empfehlungen bis hin zu Finanz- und Gesundheitslösungen, und die Zahl der Anwendungen, die tiefe neuronale Netze nutzen, nimmt weiter zu. Viele der jüngsten Forschungsbemühungen im Bereich des tiefen Lernens konzentrierten sich jedoch in erster Linie auf neuronale Netze und Bereiche, in denen sie sich auszeichnen, wie z.B. Computer Vision, Audioverarbeitung und Verarbeitung natürlicher Sprache. Die Daten in diesen Bereichen sind in der Regel homogen, während heterogene Tabellendatensätze relativ wenig Beachtung finden, obwohl sie extrem häufig vorkommen. Tatsächlich sind mehr als die Hälfte der Datensätze auf der Google-Datensatzplattform strukturiert und können in Tabellenform dargestellt werden. Das erste Ziel dieser Studie ist es, eine durchdachte und umfassende Analyse der Anwendung von tiefen neuronalen Netzen für die Modellierung und Generierung von tabellarischen Daten zu liefern. Darüber hinaus wird ein Open-Source-Benchmark für tabellarische Daten vorgestellt, bei dem wir über zwanzig maschinelle und Deep-Learning-Modelle auf heterogenen tabellarischen Datensätzen gründlich vergleichen. Der zweite Beitrag bezieht sich auf die Erzeugung synthetischer tabellarischer Daten. Inspiriert durch ihren Erfolg in anderen homogenen Datenmodalitäten werden tiefe generative Modelle wie Variation Autoencoder und generative adversarial Networks auch für die Erzeugung tabellarischer Daten verwendet. Die Verwendung von transformatorbasierten großen Sprachmodellen (die ebenfalls generativ sind) für die Generierung tabellarischer Daten wurde jedoch in der Forschung kaum beachtet. Unser Beitrag zu dieser Arbeit besteht in der Entwicklung einer neuartigen Methode zur Generierung tabellarischer Daten auf der Grundlage dieser Familie autoregressiver generativer Modelle, die bei mehreren anspruchsvollen Benchmarks die aktuellen State-of-the-Art-Methoden zur Generierung tabellarischer Daten betrifft. Ein weiterer entscheidender Aspekt für ein Deep-Learning-Datensystem ist, dass es zuverlässig und vertrauenswürdig sein muss, um in der Praxis eine breitere Akzeptanz zu finden, insbesondere in lebenskritischen Bereichen. Eine Möglichkeit, Vertrauen in ein datengesteuertes System zu bringen, ist die Verwendung erklärbarer maschineller Lernmethoden. Dennoch liefern die derzeitigen Erklärungsmethoden oft keine robusten Erklärungen, da sie sehr empfindlich auf die Wahl der Hyperparameter oder sogar auf Änderungen des Zufallsseeds reagieren. Darüber hinaus basieren die meisten dieser Methoden auf der Relevanz der Merkmale und ignorieren die entscheidende Beziehung zwischen den Variablen in einer Sample.

Das dritte Ziel dieser Arbeit ist es, diese beiden Probleme zu lösen, indem robustere und stabilere Erklärungen angeboten und die Beziehungen zwischen den Variablen durch eine grafische Struktur berücksichtigt werden.

Zusammenfassend lässt sich sagen, dass diese Arbeit einen bedeutenden Beitrag geleistet hat, der viele Bereiche berührt, die mit tiefen neuronalen Netzen und heterogenen Tabellendaten sowie der Verwendung von Methoden des erklärenden maschinellen Lernens.

Acknowledgments

First of all, I would like to express my highest gratitude and appreciation to my advisor, Prof. Dr. Gjergji Kasneci, who gave me the opportunity to pursue my goals and who believed in my abilities. Also, I thank him for guiding me along until the completion of this thesis by providing many necessary suggestions and helping me to become a researcher.

I want to extend a special thank you to Kathrin Seßler for the inspiration she provided and for our countless discussions. Having the opportunity to work with her was indeed a great fortune.

Next, my gratitude also goes to my colleagues, Tobias Leemann, Martin Pawelczyk, Johannes Haug, and Hamed Jalali. Our discussions over the years have been enlightening and I have learned immensely from all of you.

Furthermore, I am deeply grateful to Philipp Schuster for helping me identify right questions and for his unwavering support during the long and often difficult journey.

Finally, I thank my dear Mother and Father, for being and supporting me on this challenging journey in their own ways.

Contents

Abbreviations

AE	AutoEncoder
ANN	Artificial Neural Network
BERT	Bidirectional Encoder Representations from Transformers
CCPA	California Consumer Privacy Act
CNN	Convolutional Neural Network
CTR	Click-Through Rate
CPU	Central Processing Unit
CRF	Conditional Random Field
DNN	Deep Neural Network
DL	Deep Learning
DB	Data Base
FCNN	Fully-Connected Neural Network
FR	Feature Ranking
FS	Feature Selection
GAN	Generative Adversarial Network
GDPR	General Data Protection Regulation
GPT	Generative Pre-trained Transformer
GPU	Graphics Processing Unit
GSR	Galvanic Skin Response
GBDT	Gradient Boosting Decision Trees
HCI	Human-Computer Interaction
LLM	Large Language Model
MAI	Medical Articial Intelligence
ML	Machine Learning
NN	Neural Network
RF	Random Forest
RLE	Relational Local Explanations
RBM	Restricted Boltzmann Machine
RNN	Recurrent Neural Network
SMOTE	Synthetic Minority Over-sampling Technique
SSL	Self-supervised Learning
TPU	Tensor Processing Unit
VAE	Variational AutoEncoder

List of Publications

Accepted Publications

Paper 1

Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. *"Deep Neural Networks and Tabular Data: A Survey"*. In the IEEE Transactions on Neural Networks and Learning Systems Journal, December 2022. [1]
`https://doi.org/10.1109/TNNLS.2022.3229161`

Paper 2

Vadim Borisov, Klaus Broelemann, Enkelejda Kasneci, and Gjergji Kasneci. *"DeepTLF: Robust Deep Neural Networks for Heterogeneous Tabular Data."* In International Journal of Data Science and Analytics, Springer, August 2022. [2]
`https://doi.org/10.1007/s41060-022-00350-z`

Paper 3

Vadim Borisov, Johannes Haug, and Gjergji Kasneci. *"CancelOut: A Layer for Feature Selection in Deep Neural Networks"*. In the Artificial Neural Networks and Machine Learning Conference – ICANN 2019: Deep Learning: 28th International Conference on Artificial Neural Networks, Munich, Germany, September 17–19, 2019, Proceedings, Part II. Springer-Verlag, Berlin, Heidelberg, 72–83. [3]
`https://doi.org/10.1007/978-3-030-30484-3_6`

Paper 4

Vadim Borisov, Johannes Meier, Johan van den Heuvel, Hamed Jalali, and Gjergji Kasneci. *"A Robust Unsupervised Ensemble of Feature-Based Explanations using Restricted Boltzmann Machines."* At the Neural Information Processing Systems conference - NeurIPS 2021: eXplainable AI approaches for debugging and diagnosis workshop. [4]

Paper 5

Yao Rong, Tobias Leemann, **Vadim Borisov**, Gjergji Kasneci, and Enkelejda Kasneci. *"A Consistent and Efficient Evaluation Strategy for Attribution Methods"*. At the International Conference on Machine Learning - ICML, Proceedings of the 39th International Conference on Machine Learning, PMLR, 162:18770-18795, July 2022. [5]

Paper 6

Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. *"Language Models are Realistic Tabular Data Generators"*. Proceedings of the Eleventh International Conference on Learning Representations - ICLR 2023, May 2023. [6]

Manuscripts Under Review

Paper 7

Vadim Borisov and Gjergji Kasneci.*"Relational Local Explanations"*. In submission. December 2022. [7]

Applied Publications

Paper 8

Vadim Borisov, Enkelejda Kasneci, Gjergji Kasneci. *"Robust cognitive load detection from wrist-band sensors"*, Accepted to the Computers in Human Behavior Reports Journal, Volume 4, 2021, ISSN 2451-9588. [8]
`https://doi.org/10.1016/j.chbr.2021.100116`

Paper 9

Michael Gröger, **Vadim Borisov**, and Gjergji Kasneci. *"BoxShrink: From Bounding Boxes to Segmentation Masks"*. Accepted to the Medical Image Learning with Limited & Noisy Data workshop at International Conference on Medical Image Computing and Computer-Assisted Intervention MICCAI, Lecture Notes in Computer Science, vol 13559. Springer, September 2022. [9]
`https://doi.org/10.1007/978-3-031-16760-7_7`

Paper 10

Nikolaos Nikolaou, Ingo P Waldmann, Angelos Tsiaras, Mario Morvan, Billy Edwards, Kai Hou Yip, Giovanna Tinetti, Subhajit Sarkar, James M Dawson, **Vadim Borisov**, Gjergji Kasneci, Matej Petkovic, Tomaz Stepisnik, Tarek Al-Ubaidi, Rachel Louise Bailey, Michael Granitzer, Sahib Julka, Roman Kern, Patrick Ofner, Stefan Wagner, Lukas Heppe, Mirko Bunse, and Katharina Morik. *"Lessons Learned from the 1st ARIEL Machine Learning Challenge: Correcting Transiting Exoplanet Light Curves for Stellar Spots"*. In submission to the Astronomical Journal, July 2020. [10]

Chapter 1

Introduction

Various aspects of modern society are impacted by machine learning algorithms that are based on deep neural networks. There have been numerous successful applications of deep-learning-based algorithms [11, 12], they can be found in data-driven applications such as spam filtering [13], machine translation and multilingual tasks [12, 14], face recognition [15], educational systems [16], object detection [17], robotics [18], autonomous driving [19], and so forth.

Deep learning systems that are driven by data have become tremendously popular and successful due to several factors, including the availability of data, the power of computation, and the high flexibility of algorithms [12, 20, 21]. Specifically, deep models based on convolution networks [11, 22], networks with recurrent mechanisms [23] or self-attention-based Transformer networks [24] have exhibited unprecedented performance in a variety of application fields.

Most methods where deep learning algorithms that are particularly effective rely on homogeneous data, such as images, videos, audio, and text. However, heterogeneous tabular data is among the oldest forms of data for statistical analysis; it is also ubiquitous in many important real-world applications [25], such as healthcare [26], finance [27], manufacturing [28], climate science [29], and many other applications that are based on structured databases. Moreover, it is one of the leading data formats for anomaly detection [30], recommendation systems [31], cybersecurity [32], and other data-driven tasks [33, 34, 35, 36, 37].

The successful application of deep neural network models on heterogeneous tabular data has the potential to bring significant advancements to the field of machine learning. First, deep neural networks are known for their ability to capture *complex* non-linear relationships [38], providing a more comprehensive coverage of data. This can benefit the machine learning community by allowing better predictions and more accurate modeling of the underlying patterns in the data. Second, deep neural network models are able to incorporate multiple data modalities, a process known as multimodal learning [39]. It can improve the performance of data-driven systems by enabling an end-to-end learning approach that can handle diverse data types, such as visual, textual, and tabular data. Lastly, deep neural network models provide a range of robust learning strategies with limited labels, including self-supervised learning [40], transfer learning [41], and synthetic data generation [1]. These methods can facilitate effective learning on heterogeneous tabular data, even in the absence of large amounts of labeled data.

The generation of synthetic tabular data is another area where deep neural networks have a crucial role to play [42, 1]. Synthetic tabular data is artificially generated by algorithms data instead of being collected from real-world sources. This data is useful in numerous applications, such as testing and evaluating machine learning algorithms, training machine-learning algorithms, and protecting sensitive information [43]. Deep neural networks are particularly well suited for generating synthetic tabular data because they can learn to synthesize data that has the same statistical properties as real-world data, allowing them to produce data that is similar to what would be collected from real-world sources. As a result, deep learning approaches are an important tool for synthesizing tabular data.

Lastly, even though deep neural networks are showing wide adoption in various areas, one of the main challenges in using deep learning data systems is their lack of explainability [44]. This makes it difficult to understand how the model arrives at its predictions, which can limit the users' trust in the results.

To address this issue, researchers and developers are exploring the use of explainable machine-learning techniques [44, 45, 46, 47, 48, 49]. These techniques aim to make the inner workings of deep neural networks more transparent, allowing users to better understand the basis for the model's predictions. Hence, they help to increase trust in the model and provide valuable insights into the data and the underlying relationships that the model has learned.

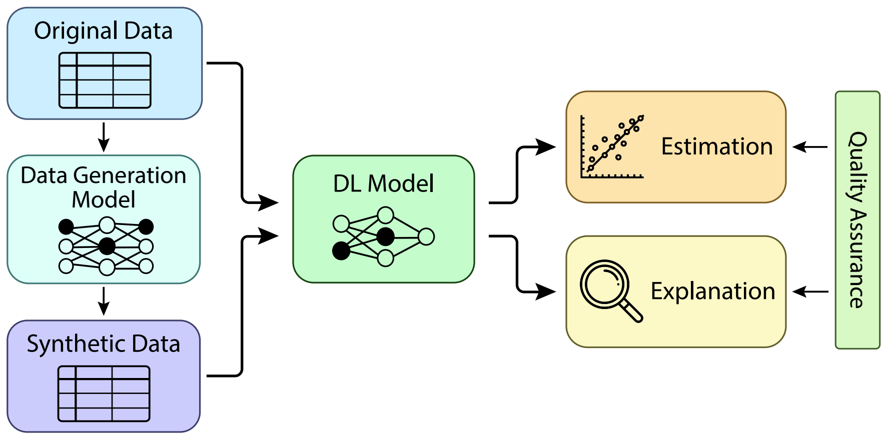


Figure 1.1: A typical data pipeline for a machine-learning application. The original data is preprocessed and augmented with synthetic data. This enhanced dataset is then fed into a machine learning model that generates predictions based on the input data. These predictions can be further explained through explanation techniques, which provide insights into the decision-making process of the machine learning model. This allows for a deeper understanding of the model's behavior and potential limitations. In general, synthetic data and interpretable models contribute in improving the accuracy and transparency of machine-learning applications. Note this is a simplified version of a data pipeline.

1.1 Main Objectives

The successful deployment and operation of data-driven applications using deep neural networks on tabular and other data modalities require numerous tasks (Fig. 1.1), among which we identified *three core challenges*, which are the *central subjects* of the work:

1. **Tabular Data Modeling.** An imperative aspect of data analysis is inference, which involves making estimation based on existing observational data. In spite of the fact that a powerful and robust predictive model is crucial for all applications, the interaction between tabular data and deep neural networks transcends beyond mere inference. It has been noted by numerous independent studies [1, 50, 51, 52] – in comparison to decision tree ensemble-based methods, such as gradient boosting [53], artificial neural networks demonstrate inferior performance on heterogeneous tabular data.

2. **Synthetic Tabular Data Generation.** It has been demonstrated that deep neural networks can benefit from large labeled or unlabeled datasets utilizing self-

supervised or unsupervised learning methods [54, 55]. However, the data acquisition is costly and frequently unavailable to the end user [56, 57]. In fact, it is even more crucial in areas such as medical machine learning, where the price of new data samples is enormous [58] and the data is often imbalanced. Apart from collecting more data, synthetic data can be used to address the issue. Furthermore, data privacy is an open issue in many critical fields, e.g., finance or medical, where artificially generated data can be used to avoid sensitive information data leakage [59].

3. **Explainability of Machine Learning Models.** As a result of strict data protection laws [60, 61] such as the California Consumer Privacy Act (CCPA) [62] and European General Data Protection Regulation (European Union [EU] GDPR) [63], which both mandate *the right to explain* for automated decision systems, interpretability is becoming a key aspect for predictive models used for tabular data [48, 64]. In addition to this, interpretability methods can also be valuable tools for model auditing and debugging during deployment [65].

The following Sections provide a detailed overview of the selected core tasks of this work. In Section 1.2 key definitions are described.

1.2 Definitions

This Section is devoted to providing definitions of key terms used in the dissertation, which are central components of the project. For more detailed explanations of the methods discussed, we also refer you to the original works.

The key concept in this dissertation is a **(deep) neural network**. Unless stated otherwise, we use this concept as a synonym for feed-forward networks and artificial neural network. As described by [12], a deep neural network defines a mapping $\hat{f}$,

$$y = f(\boldsymbol{x}) \approx \hat{f}(\boldsymbol{x}; W), \tag{1.1}$$

that learns the value of the model parameters W (i.e., the “weights” of a neural network) that results in the best approximation of the true underlying and unknown function, f. In this case, x is a multi-dimensional data sample (i.e., $\boldsymbol{x} \in \mathbb{R}^n$) with corresponding target y (where typically, $y \in \mathbb{R}^k$ for k classes and $y \in \mathbb{R}$ for regression tasks) from a data set of tuples $\{(x_i, y_i)\}_{i \in \mathcal{I}}$. The network is called feed-forward if the input information flows in one direction to the output without any feedback connections.

An important aspect for this dissertation deep neural networks architecture is the **Transformer network**, which is commonly referred to as the “foundational model” in the literature [66]. It relies on self-attention mechanisms to capture the dependencies between different input elements. By learning complex relationships between input elements using self-attention [24], also known as intra-attention [67], the model is able to focus selectively on different parts of the input. The Transformer model can be used for inference and generative tasks.

Formally, the self-attention mechanism can be represented as:

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{1.2}$$

where Q, K, and V are the query, key, and value matrices, respectively, and d_k is the dimensionality of the key vector. The self-attention mechanism produces a weighted sum of the value vectors, with the weights determined by the dot product between the query and key vectors. In this way, the model can focus on various parts of the input depending on the relative importance of each element.

This thesis primarily addresses two main types of data: **homogeneous** and **heterogeneous** data, with deeper focus on the latter data type. Let $\mathcal{D}$ be a dataset with N data points, where each data point $x_i \in \mathcal{D}$, for $i = 1,2\ldots,N$, is described by a set of features (variables) $F = \{f_1, f_2, \ldots, f_M\}$. We assume that each feature f_j takes values in a domain $Dom(f_i)$.

The first type, **homogeneous** data, includes modalities such as images, audio, or text data where only a single feature type is present. Specifically, a dataset $\mathcal{D}$ is said to be homogeneous if all the features share the same domain, i.e., the domains of all features are identical.

Definition 1 (Homogeneous Dataset) *A dataset $\mathcal{D}$ is homogeneous if:*

$$\forall (f_j, f_k) \in F \times F, \ \textit{such that } Dom(f_j) = Dom(f_k). \tag{1.3}$$

The second type, **heterogeneous data**, typically contains a variety of attribute types. These include both continuous and discrete attributes of different types (e.g., binary, ordinal, and low- and high-cardinality categorical values). A dataset $\mathcal{D}$ is said to be heterogeneous if there exists at least one pair of features $(f_i, f_k) \in F$, with $j \neq k$, such that the domains of these features, i.e., $Dom(f_i) \neq Dom(f_k)$.

Definition 2 (Heterogeneous Dataset) *A dataset $\mathcal{D}$ is heterogeneous if:*

$$\exists (f_j, f_k) \in F \times F, \ \textit{such that } Dom(f_j) \neq Dom(f_k). \tag{1.4}$$

Categorical variables represent an attribute type of significant importance [68, 69]. According to the definition given by Lane [70], categorical variables are qualitative in nature. They “do not imply a numerical ordering,” unlike quantitative values, which are “measured in terms of numbers.” Typically, a categorical variable can assume one of a finite set of values. Examples of common categorical variables include `gender`, `user_id`, `product_type`, and `topic`.

Tabular data, also known as *structured data* [71], is a type of heterogeneous data that is typically presented in a table format with rows representing data points and columns representing features. In this context, we consider a data set to be tabular if it has a fixed number of features that can be either continuous or categorical. Each data point can be viewed as a row in the table, or alternatively, as a sample from the underlying distribution. An illustrative example of five rows of a heterogeneous, tabular data is provided in Table 1.1. Tabular data is widely used in various fields [1], including finance, marketing, and healthcare, to analyze and make predictions based on the relationships between the features.

Synthetic tabular data, on the other hand, is a type of data that is generated using an algorithm or several methods, rather than being collected from real-world sources. It is structured in a way that is similar or ideally not distinguishable from original real-world data, with rows and columns that represent different variables and observations. However, because it is generated artificially, it can be customized and controlled in ways that real-world data cannot be, e.g., conditionally sampled.

There are several reasons why synthetic tabular data can be used, for instance, testing and evaluating machine-learning algorithms. This is because synthetic data can be generated with known properties and characteristics that allows practitioners and researchers to easily control the data and test their algorithms in a more controlled environment. Another reason for using synthesized tabular data is that it can be useful for training machine learning algorithms. As an example, if real-world data is not available or is difficult to obtain, synthetic data may be used to train the algorithm and help it become more accurate. It is also possible to generate synthetic data in a manner that does not include sensitive information, making it useful for privacy and security purposes [72].

Explainable machine learning. The complexity of current machine learning algorithms, particularly deep-learning-based algorithms, makes it difficult for humans to comprehend the exact decision making process that occurs during inference. These aim to make the internal workings of these algorithms more transparent and their outputs easier to interpret.

In machine learning, an explainable model or algorithm is one that provides a clear and interpretable explanation for its predictions or decisions. An explanation can be provided in natural language, a visual representation of the model's decision-making process, or other methods of illustrating the model's logic. Explainability can also be defined as the degree to which a model's predictions can be understood or justified by a human being. It may be necessary to assess both the transparency of the model's structure and the interpretability of its output, as well as the reliability of the explanations it provides.

In the literature, there are several definition of explainability. For example, according to Miller [73] "Interpretability is the degree to which a human can understand the cause of a decision." Kim et al. [74] propose the following definition for explainability "the degree to which a human can consistently predict the model's result." As a general rule, explainability is critical to machine learning because it helps to establish trust in the model, increase its accountability, and allow it to be used in real-world situations. This information may also help users identify potential areas for improvement or further investigation, as well as understand the machine learning model's limitations and potential

Age	Education	Occupation	Sex	Workclass	Income
39	Bachelors	Adm-clerical	Male	Private	$\leq$50K
50	Bachelors	Exec-managerial	Male	Private	$>$50K
38	HS-grad	Handlers-cleaners	Male	Private	$\leq$50K
53	11th	Handlers-cleaners	Male	?	$\leq$50K
28	Bachelors	Prof-specialty	Female	State-gov	$>$50K

Table 1.1: An example of a heterogeneous tabular data set. Here we show five random samples with selected variables and label columns (Income). from the Adult income data set [77]. As it can be seen, the displayed data samples are highly heterogeneous, and contain missing values (?).

biases. Also, it provides a better understanding of the data.

Lastly, in the field of explainable machine learning, understanding, interpreting, and explaining are often used interchangeably [75, 76]. Interpretability typically refers to the ability to comprehend the overall functioning of a prediction model, while explainability involves providing explanations that are not necessarily understandable by the model itself. This work uses explainability as a synonym for interpretability, unless stated otherwise.

1.3 Deep Neural Networks and Tabular Data

The use of deep neural networks has been proven to be effective in a variety of areas, including image recognition, natural language processing, and even the prediction of stock prices. However, when it comes to dealing with heterogeneous tabular data, these networks face challenges [1, 52, 25, 51]. In this context, decision-tree-based approaches, such as random forest [78] and gradient boosting decision trees [79], have been found to be superior to deep neural networks. These methods take advantage of the structured nature of tabular data and are able to effectively represent complex relationships between various variables. As such, they have become the state-of-the-art approach for dealing with this type of data.

Yet, in real-world applications, the most common and one of the oldest data type is heterogeneous tabular data [25, 80, 81]. According to the Google Dataset Search platform – over 65% of data sets there contain tabular files in either CSV or XLS formats [82]. Kadra et al. [83] called tabular data sets the “last unconquered castle” for deep neural network models.

One of the major disadvantages of decision tree-based methods is the need to store almost the entire dataset in memory during the training process [84]. This can be a significant computational burden, particularly for large datasets. In addition, decision tree-based models are not well-suited for multimodal datasets that include different data

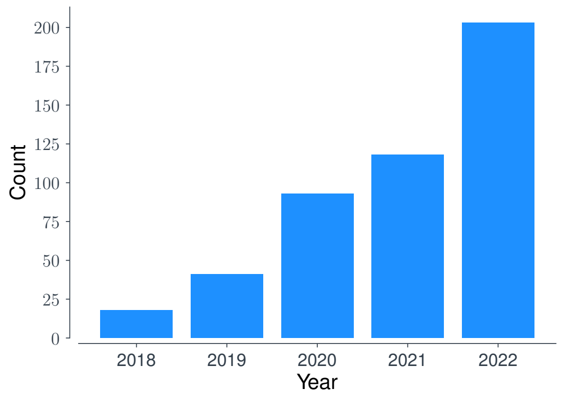


Figure 1.2: Estimated counts of annually published paper on deep neural network models for tabular data modeling or generation. This field has been experiencing rapid growth since 2019. We collect the data utilizing the `dimensions.ai` web-platform with the query: *deep neural networks and tabular data*.

types, such as visual and tabular data. In contrast, deep neural networks can be trained using batch learning, which does not require the entire dataset to be stored in memory. Deep learning methods have been shown to provide state-of-the-art performance on multimodal data tasks, making them a superior choice in these cases [39].

Over the past years, specialized deep learning approaches for tabular data have grown in popularity and have been applied in various fields (Fig.1.2), including life-critical areas such as medicine [1, 42]. The development of dedicated tabular data deep learning techniques has allowed for more scalable and fast inference. Consequently, deep learning approaches to tabular data are becoming an essential tool in many fields and are likely to continue to play an important role in data analysis for some time to come.

Despite the theoretical benefits of deep neural networks, their application to heterogeneous tabular data may pose challenges due to the lack of spatial information, missing values, a mixture of feature types (numerical, ordinal, and categorical); and a lack of prior knowledge of the dataset structure. Compared to classical decision tree algorithms, which are known for their transparency, this lack of transparency or interpretability is a significant disadvantage. Additionally, the complexity of deep neural networks can make them difficult to maintain, since the deep learning methods usually require more time for the hyper-parameter and significant computational resources.

To overcome these challenges, in recent years, numerous work have been proposed [1]. We group these approaches into three categorizes: data encoding methods, specialized architectures, and regularization models.

Data encoding methods transform categorical and numerical data in order to enable deep neural network models better extraction of the information signal. These methods do not require new architectures or adaptations of the existing data processing pipeline. However, the transformation step comes with an increased preprocessing time, which may be a concern for high-load systems [85], particularly in the presence of categorical variables with high cardinality and growing data set sizes. Single-dimensional encodings and multi-dimensional encodings are two subcategories within this group. Single-dimensional encodings transform each feature independently, while multi-dimensional encodings map an entire record to another representation.

Specialized architectures are the most commonly investigated approach for deep learning on tabular data [1]. These architectures suggest that a different deep neural network architecture is required for tabular data. For example, hybrid models, which fuse classical machine learning approaches (e.g., decision trees) with neural networks, and transformer-based models, which rely on attention mechanisms [24], are two important types of specialized architectures.

Regularization models propose that the moderate performance of deep learning models on tabular data is due to their extreme sensitivity and model complexity. Therefore, strong regularization schemes are suggested as a solution, implemented mainly in the form of special-purpose loss functions. Overall, these approaches offer a range of solutions for deep learning on tabular data, each with their own benefits and drawbacks.

1.4 Tabular Data Generation

There is a lot of cost involved in collecting data, and often new data cannot be obtained by the user [86]. Therefore, the researches is constantly looking for a new methods for realistic data generation. Whereas for visual and textual data the generation of realistic samples are very well studied using variational autoencoders (VAEs) [87], generative adversarial networks (GANs) [88], and denoising diffusion probabilistic models [89], tabular data is still a challenge.

One of the main advantages of synthetic tabular data is that it can be used to augment or replace real-world data for various purposes [90], such as testing algorithms or training machine learning models. Another benefit of synthetic tabular data is that it can be used to generate data with specific characteristics [91], such as a certain distribution of values or a specific level of noise, applying the conditional sampling. Research can benefit from testing algorithms or models under various conditions in order to gain a better understanding of how they will perform under different circumstances.

With that said, synthetic tabular data provides researchers and practitioners with the ability to generate custom data that can be used for a variety of purposes. As a result, it is possible to reduce the amount of required real-world data, and it is possible to provide more controlled and predictable data for testing and experimentation.

In numerous real-life applications, the generation of realistic tabular data is fundamental. Three of the major purposes are data augmentation [92], data imputation (i.e., the filling of missing values) [93, 94] and class balancing [95, 96, 97, 98]. An additional subject is privacy-aware machine learning [43, 99, 100] where the generated data can potentially be leveraged to overcome privacy concerns.

1.4.1 Challenges of Tabular Data Generation

There are several challenges associated with the generation of realistic tabular data:

- **Mixed data types.** The first issue is the heterogeneous nature of the tabular data, e.g., it structured from different discrete and continuous distributions (Sec. 1.2). In order to synthesize different modalities, deep generative models need to be adapted to each dataset's architecture. Also, Srivastava et al. [101] showed that a vanilla generative adversarial network (GAN) could not model all modes on artificial 2-dimensional datasets.

- **Often non-Gaussian distributions.** Usually, generative modals utilize the assumption that variables are distributed normally, e.g. pixel values. In the case of the heterogeneous tabular data, features do not always follow this distribution.

- **Categorical data.** The probabilistic modeling of categorical data is one of the most critical issues of tabular data generation [102, 103], since categorical variables can have high cardinality numbers. Furthermore, categorical features from real-world tabular datasets are regularly imbalanced, which makes it harder to model.

- **Semantic coherence between the variables.** Even thought there is no connection between variables in tabular data as there is in textual or visual, the values in a sample can have semantic coherence, e.g., a data sample contains two variables: `"Car brand"` and `"Car brand origin."` And if the generated data has this form `"Car brand"` is `"Mercedes"` and `"Car brand origin"` is `"Italy,"` we conclude that there is a semantic mistake.

- **Tabular data quality.** Last but not the least, real-world tabular data datasets often suffer from the low-quality data problem [1]. It is common for tabular datasets to contain missing values [104], extreme data (e.g., outliers) [105], erroneous or inconsistent data [106], and to have a small size relative to the high-dimensional feature vectors that are generated from the data [107]. Moreover, due to the cost of data collection, tabular data tend to be class-imbalanced. Lastly, it has been showed that artificial neural networks are less prone to low-quality data issues, than decision-tree-based methods [1].

  All of these challenges need to be addressed in order for deep neural networks to be able to produce realistic models.

As of now, the most common method of generating synthetic data is through the use of statistical models [108, 109] or generative deep neural networks [86]. Traditionally, deep generative models perform well on continuous features, while Bayesian networks perform well on categorical features.

One of the most popular statistical methods for the tabular data generation step is synthetic minority over-sampling technique (SMOTE) method [110].The SMOTE algorithm selects an instance of a minority class at random and locates *k* of its nearest minority class neighbors. Other approaches from this group include Bayesian networks and multivariate cumulative distribution functions, e.g., copula models.

In recent years, inspired by the success in homogeneous data generation, approaches based on deep neural network are rising for tabular data synthesis. These are based on mostly deep learning architectures: generative adversarial networks (GANs) [88] and variational autoencoders (VAEs) [87]. Among the most popular and widely used deep learning models for generating tabular data is conditional tabular GAN (CTGAN) [102]. The same authors of the CTGAN model also proposed tabular VAE (TVAE).

1.4.2 Evaluation of Synthetic Tabular Data

In recent years, numerous measures for evaluation of synthetic data have been proposed. These measures can be divided into two groups: single-dimensional and multidimensional scores. Whereas the first group compare each variable individually, the latter approaches combine multiple features together.

Besides the dimensional differences of the approaches, they are also distinct in an algorithmic manner. For example, one group of approaches uses the idea that the synthetic data should be "similar" to the original distribution, whereas the second group uses surrogate models as a means to differentiate between the original and synthetic data.

In general, the evaluation scores can be categorized into three broad categories:

- **Statistical scores.** These scores evaluate synthetic tabular data by performing statistical tests on them. Some compare multiple columns simultaneously, while others compare individual columns individually and then provide a combined result. Another option is to use likelihood metrics, which try to fit a probabilistic model to the actual data and then assess the likelihood of the synthetic data based on it.

- **Detection measure.** The central idea of this method is to utilize another machine learning model to distinguish between real and synthetic samples, after the chosen performance (e.g., accuracy, F1, ROC-AUC) score is reported. Machine learning efficacy is one of the most adopted measures.

- **Privacy measure.** The last set of scores measures the privacy-related metrics. These metrics test the accuracy of an adversarial attacker model on synthetic data and then evaluate the model's accuracy on real data.

The choice of evaluation measures for synthetic data depends on the specific characteristics of the data and the goals of the evaluation. The strengths and weaknesses of both single-dimensional and multidimensional measures should be considered when selecting the appropriate measure based on the specific context and requirements of the evaluation.

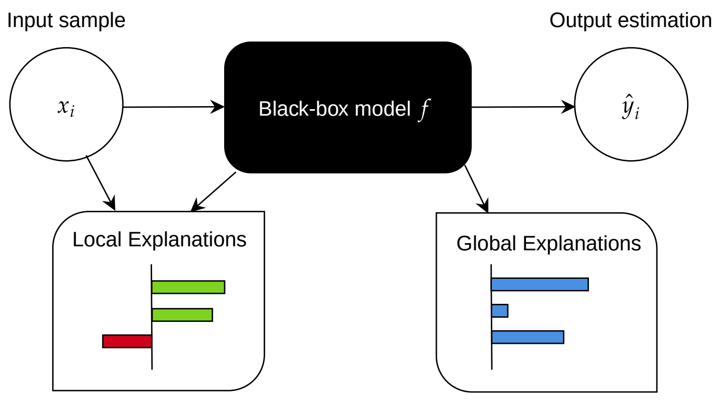


Figure 1.3: An illustration of local and global explanations for a black-box model f given a data point of interest x_i.

1.5 Explainability of Deep Neural Networks

Recent research studies have drawn considerable attention to the problem of explaining machine and deep learning models [44, 45, 46, 47, 48, 49]. There are numerous reasons for this: in visual domains, deep convolutional neural networks have shown superior performance for various tasks [111]: object detection [17], segmentation [112], classification [113], and other areas. In the natural language processing domain, self-attention models, specifically Transformer models [24] constitute the state of the art for many applications such as text summarization [114], sentiment analysis [115], and so forth.

Despite the fact that deep learning algorithms have performed well in many real-world applications, without trust in the data systems, in addition to high-risk decision tasks, it may be difficult to implement them in the future. As a response, the understanding that the machine learning model performs as planned can be observed from the explanations. In this manner, explainable machine learning can be used as a tool for model debugging.

Furthermore, a data system user can also enhance the outcome by learning the reasons behind the specific way the given machine learning model performs, after obtaining model's explanation [116]. A machine learning model's explanation can be used to identify new data patterns, which might otherwise be invisible to the human eye. The last thing we ought to mention is that, there have been government regulations made regarding the use of the data-driven algorithms in decision-making systems, e.g., European Union (GPDR), California Consumer Privacy Act (CCPA) [62, 63].

In regards to the data modalities, for visual data, saliency maps are usually used by highlighting the most crucial areas. The explanation of tabular data requires additional efforts [48], such as a handling the categorical variables. Additionally, the use of visualizations, such as bar graphs and scatter plots, can aid in the interpretation of tabular data. Furthermore, the use of natural language explanations is another possibility to explain the results of machine learning models [117]. These explanations should be concise and easy to understand, highlighting the key features and patterns in the data that led to the model's estimation.

Moreover, an aggregation of feature attribution metrics [118, 119] can be used for the global explanations of a machine learning model. The main drawback of this approach is that the aggregation step is then required significant amount of time, since every local explanation are generated individual.

We discuss and formally define two common approaches for obtaining post-hoc explanations of machine learning models in the following sections - local and global explanations.

1.5.1 Local and Global Explanations

Overall, there are two major approaches in providing explanations of a given machine learning model: local explanations and global explanations [120]. The main difference between them is that the local explanation provides an explanation on a single data sample, whereas the global explanations usually display the overall importance of several or all available data points.

We will discuss each of these types in the following sections. It should be noted that there are other forms of explanation, including counterfactual [121], prototypical [122], conceptual [123], which are beyond the scope of this dissertation.

Local Explanations

Definition 3 (Local Explanation) *A local explanations (feature attribution[1]) function can be seen as $\phi(f, \boldsymbol{x}, c_{\boldsymbol{x}}) \in \mathbb{R}^n$, where $f : \mathbb{R}^n \to \mathbb{R}$ is a black-box model and $\boldsymbol{x} \in \mathbb{R}^n$ is an input sample belonging to a class $c_{\boldsymbol{x}} \subset \mathbb{R}$. The output of ϕ is an explanation representation vector $\mathbf{e}_{\boldsymbol{x}} \in \mathbb{R}^n$.*

Each element of $\mathbf{e}_{\boldsymbol{x}}$ is an importance score assigned to a feature value in $\boldsymbol{x}$. A large positive or negative value in $\mathbf{e}_{\boldsymbol{x}}$ signifies that a particular feature greatly influences the outcome. Features with near-zero values in $\mathbf{e}_{\boldsymbol{x}}$ have little impact. Note that there are explanation methods that do not require a class specification; thus, for a simpler and more general notation, a feature scoring function has the form $\phi(f, \boldsymbol{x})$.

A vast amount of literature has been dedicated to the local explanation of deep neural networks in recent years [124]. The local-explanations approaches can be broadly separated into three groups: gradient-based, attribution, and perturbation-based methods. There are other approaches for feature attribution analysis [49, 125]; however, in this dissertation, we concentrate on the three main groups.

[1]Unless stated otherwise we use the *local explanation* definition as a synonym for *feature attribution* concept.

Gradient-based methods. These approaches utilize the gradient of the loss with respect to the information on every layer's input [126, 127]. The idea is to compute the gradient, or an approximation of it [128, 129], for the input image. Thus, we get knowledge about how the class information changes with respect to a slight variation in the input image. The positive gradients show that a small change to that input variable may increase the output probability; the negative gradient shows the opposite behavior. However, small changes in the input may lead to a strong increase or decrease of the output probability for a given class. Normally, these methods' output is the same shape as the image, which is easy to interpret and presents a network's "area of attention" view.

Gradient-based explanation methods for deep neural networks are powerful and scalable, but they often generate noisy explanations [130] and indicate only the key pixels that mark the rest as neutral or unimportant. The saturation problem [131] also limits and misleads the explanation of the gradient-based methods. Many of these methods require a baseline input, which can significantly affect the explanation outcome [132]. Also, a recent work [133] suggests that input gradients do not highlight discriminative features. Furthermore, it has been shown that gradient-based methods are not robust [134]. Another limitation of these approaches is that they are weak in the estimation of trust or justification, such as estimating uncertainties of the importance score.

Attribution methods. These approaches decompose, in a recursive manner, the decision made by a deep neural network. A layer-wise relevance propagation (LRP) [135] method works by propagating the prediction backward in the neural network, according to a set of rules that have been deliberately designed. There are several extensions to LRP work suggested in recent years, such as contrastive-LRP (CLRP) [136] and Softmax-Gradient-LRP (SGLRP) [137]. They solve the main disadvantage of the LRP method, the class agnostic behavior. DeepLIFT [131] compares each neuron's activation to its reference activation and assigns contribution scores according to the difference. The work [138] provides theoretical insights on the limitations of these methods. While the attribution methods are moderately fast and produce an accurate explanation, they do not quantify saliency maps with uncertainties. Hence, it is challenging to estimate the confidence of the feature importance produced by these methods, bringing more trust in the system.

Perturbation-based methods. The next group of approaches utilizes a simple but reliable method to generate the local explanation. Perturbation-based methods [139] rely on the idea that if we change the input image, we will find which areas are critical for the artificial neural network under test. Thus, these methods are not based on backpropagation. The most notable algorithms in this category are the Local Interpretable Model-Agnostic Explanations (LIME) [118] and SHapley Additive exPlanations (SHAP) [119] algorithms. SHAP approximates for each feature a Shapley value. Both LIME and SHAP have a substantial theoretical justification [140, 141, 119]. Another group of methods uses the random masking approach [142, 143].

The main drawback of these algorithms is that the data perturbation is an enormous computational complexity task, especially for high-denominational data, as it is frequently the case in computer vision tasks. Therefore, LIME and SHAP utilize the superpixel strategy for image data. The idea is to arrange pixels on the image into groups called superpixel and use them as a discrete feature. Such an arrangement significantly lowers the number of perturbations needed. LIME splits the image to explain in contiguous patches that share color and or brightness similarities, SHAP divides the image into rectangles ignoring the shapes and colors. However, it has been shown [129, 144] that superpixel-based explanations are not robust and highly depend on the chosen method of superpixel generation.

Global Explanations

Definition 4 (Global Explanation) *A global explanations (feature selection) function can be seen as $\xi(f, \mathcal{D}) \in \mathbb{R}^n$, where $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is a black-box model and $\mathcal{D}$ is a data set of tuples $\{(x_i, y_i)\}_{i \in \mathcal{I}}$, where $\mathcal{I}$ is the total number of samples. The output of ξ is a global feature importance vector $\mathbf{g}_f \in \mathbb{R}^n$.*

The elements of $\mathbf{g}_f$ represent feature importance values for each variable in a dataset $\mathcal{D}$ given a machine learning model f. Note the feature selection task [145, 3] can be seen as global explanation extraction [146], since the output of global explanation function is feature importance scores. Feature selection is a process for dimensionality reduction where the purpose is to remove noisy or not-important input variables by identifying the feature importance.

In the recent years plenty of approaches were proposed for the global explanations [47]. Usually, these methods utilizing learned parameters for feature scoring procedure. In case of the decision-tree-based algorithm, features that used the most for data splitting are ranked higher [147]. For deep neural network models, different ways of assessing global explanations exist; we identify three major groups of approaches: perturbation-based methods, regularization-based techniques, special-architecture approaches. Lastly, an aggregation of local explanation, can be also used for the global explanation task.

For the global explanation of deep learning models (as for local explanation) has numerous approaches. The most common technique is the permutation method; typically, it involves replacing or removing a variable from the entire training dataset to analyze changes in the model output. This method allows for the determination of the value of a given feature, but it is computationally intensive, requiring the training of a deep learning model n times to evaluate n features. Additionally, the permutation method only provides an approximation of the importance of a feature as it does not consider the interactions between features or the overall structure of the model.

Alternative methods such as trained parameters analysis (e.g., gain split information for decision trees) or feature attribution methods not only can provide more accurate explanations, but may also be computationally expensive and may not be feasible for large and complex deep learning models.

Lastly, the explanation of deep learning models remains an active area of research [148], with ongoing efforts to develop more efficient and effective methods in interpreting these complex algorithms.

1.5.2 Evaluation of Local and Global Explanations

Due to the absence of the ground truth for local or global explanations. It is crucial to address the following research question: *How to evaluate obtained interpretations?*

In order to assess different attribution methods, an ablation approach is commonly used [149, 150, 151, 152]. It involves perturbing the most or least significant input features, such as the pixels of an image or variable in a tabular dataset. As a general rule, perturbing pixels that have high predictive quality should decrease the confidence of the machine learning model, whereas perturbing pixels with low predictive quality should have minimal impact.

According to Doshi-Velez and Kim [152], evaluation methods for explanation algorithms can be organized using two major groups: *human-grounded scores*, relying on human judgment, and *functional-grounded scores*, which utilize a dedicated function for the evaluation step.

Human-grounded evaluation scores are challenging to obtain due to the inherent cost of gathering human judgments and the subjectivity involved in evaluating explanations. They are, however, considered a reliable measure of the quality of an explanation, since they reflect the perceptions and understanding of users. Some examples of human-grounded scores are user surveys, user feedback, and expert evaluations [153].

Functional-grounded scores, on the other hand, are easier to obtain and less subjective, but they may not accurately capture the quality of the explanation from a human perspective. These scores are based on specific measures and functions that are designed to evaluate an explanation based on certain criteria, such as accuracy, comprehensibility, or relevance. Examples of functional-grounded scores include fidelity metrics [151], interpretability metrics, and coherence metrics [76].

As a result, the choice of the evaluation method for explanation algorithms depends on the specific goals and constraints associated with the application in question. The use of human-grounded scores may be more appropriate for applications that prioritize user satisfaction, while the use of functional-grounded scores may be more appropriate for applications that require fast and objective evaluation.

1.6 Contributions

The main focus of this thesis is on the development and evaluation of advanced forms of deep neural networks for heterogeneous tabular data, as well as the application of machine learning models for explainability tasks. Our research encompasses a spectrum of issues, including the limitations of current deep learning approaches on tabular data, the potential of using pretrained language models for synthetic data generation, and the need for robust explanations of machine and deep learning models.

Our main contributions to the field include the following:

- A comprehensive survey and taxonomy of state-of-the-art deep learning approaches for heterogeneous tabular data, as well as the development of an open-source evaluation framework for these methods. Additionally, we propose a novel deep tabular learning algorithm (DeepTLF) for robust learning on tabular and mixture data modalities.
- A novel approach for the generation of realistic tabular data using pretrained large language models, coined GReaT, which outperforms existing methods by a significant margin on multiple challenging experiments.
- Contributions toward reliable and robust explanations of machine and deep learning algorithms through the development of novel approaches such as relation local explanations, robust aggregation of feature attributions, and a special layer for deep neural networks for global explanations – CancelOut.

We believe that our contributions will be of great value to researchers and practitioners working in the fields of deep learning with tabular data and the explainability of machine learning models. Since our research provides valuable insights into the potential benefits and limitations of using deep neural network models for learning on heterogeneous tabular data, the findings of this study will help guide future research in this area and assist in developing more effective and robust deep learning approaches for tabular data.

Along with this thesis, we also submit two publications on applied topics, such as cognitive load estimation using machine learning methods and weakly-supervised segmentation for medical machine learning. These contributions have been published in an international conference and a scholarly journal.

1.7 Outline

The structure of this dissertation can be summarized as follows: In Chapter 2, we provide the motivation and key findings for each component of this study. A discussion of the selected topics is provided in Chapter 3 before we conclude in Chapter 4. Appendix A contains manuscripts included in this thesis.

Chapter 2

Main Findings

This Chapter summarizes the papers written during my doctoral studies on deep neural networks for tabular data and the explainability of machine learning methods. It includes motivations, methodologies, and findings according to the open directions that are presented in Chapter 1. The discussed papers are available in Appendix A.

2.1 Deep Tabular Learning

This Section is based on two journal articles: *"Deep neural networks and tabular data: A survey"*, accepted to the IEEE Transactions on Neural Networks and Learning Systems journal, and *"DeepTLF: Robust deep neural networks for heterogeneous tabular data"* published at the International Journal of Data Science and Analytics.

The Section is organized as follows: each of the following subsections represents the corresponding publication, containing a detailed discussion of the motivation and methods employed, followed by a summary of the main findings.

2.1.1 A Comprehensive Survey on Deep Neural Networks and Tabular Data

Despite the fact that deep neural networks are the current state-of-the-art approach in many fields that involve visual, audio, or text data, they tend to perform worse than decision tree ensembles when it comes to heterogeneous, tabular data [78, 53, 154, 155, 156, 157]. Nevertheless, in recent years, numerous deep learning approaches have been proposed specifically for tabular data [1]. In spite of this, there has not been a systematic and comprehensive review of these methods in order to assess their effectiveness. As a result, the primary motivation of this manuscript is to provide a thorough survey of existing research, conduct an empirical comparison of the research, and facilitate further progress in the field itself. For a complete text version of the manuscript, please refer to the Appendix A.1.

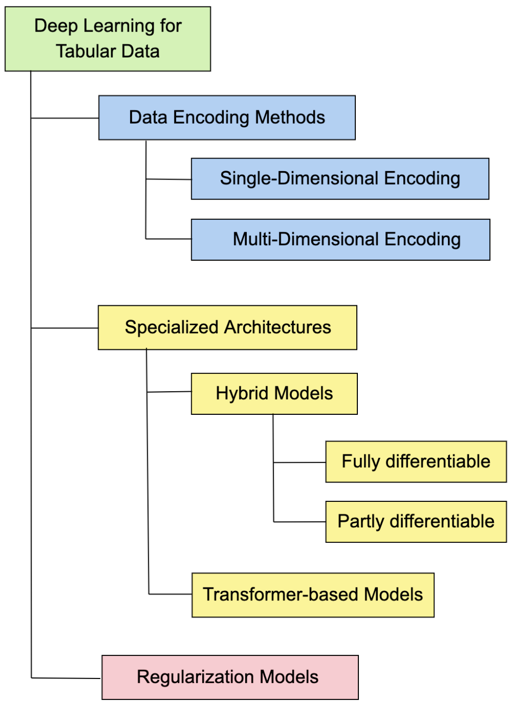


Figure 2.1: The proposed unified taxonomy of deep neural network models for heterogeneous tabular data inference tasks.

State-of-the-art deep tabular learning and modeling

There has been an increase in interest in developing novel deep learning approaches devoted to heterogeneous tabular data in recent years [1]. To the best of our knowledge, we are the first who categorize these methods into three major groups: data transformations, specialized architectures, and regularization models.

For each of these groups, our work offers a comprehensive overview of the main approaches. Figure 2.1 presents the final taxonomy of deep learning model for tabular data. In total, we thoughtfully examined twenty-three approaches for deep tabular learning. Additionally, we have developed an open benchmark to measure the performance of machine learning algorithms, which will be discussed in greater detail in the following subsection.

Moreover, we present a detailed overview of twenty-three deep learning approaches for synthetic tabular data generation (Appendix A.1). We have also included a section devoted to the assessment of generative quality of tabular data, in which we discuss the most common existing approaches.

Furthermore, we provide an overview of strategies for explaining deep models on tabular data. Discussing the current approaches and strategies. Notably, the models that build on Transformer-based approaches such as TabNet [158], TabTransformer [159],

	Method	HELOC		Adult		HIGGS		Covertype		Cal. Housing
		Acc ↑	AUC ↑	Acc ↑	AUC ↑	Acc ↑	AUC ↑	Acc ↑	AUC ↑	MSE ↓
Machine Learning	Linear Model	73.0±0.0	80.1±0.1	82.5±0.2	85.4±0.2	64.1±0.0	68.4±0.0	72.4±0.0	92.8±0.0	0.528±0.008
	KNN [162]	72.2±0.0	79.0±0.1	83.2±0.2	87.5±0.2	62.3±0.1	67.1±0.0	70.2±0.1	90.1±0.2	0.421±0.009
	Decision Tree [163]	80.3±0.0	89.3±0.1	85.3±0.2	89.8±0.1	71.3±0.0	78.7±0.0	79.1±0.0	95.0±0.0	0.404±0.007
	Random Forest [78]	82.1±0.2	90.0±0.2	86.1±0.2	91.7±0.2	71.9±0.0	79.7±0.0	78.1±0.1	96.1±0.0	0.272±0.006
	XGBoost [155]	83.5±0.2	92.2±0.0	87.3±0.2	92.8±0.1	77.6±0.0	85.9±0.0	**97.3±0.0**	**99.9±0.0**	0.206±0.005
	LightGBM [156]	83.5±0.1	92.3±0.0	**87.4±0.2**	**92.9±0.1**	77.1±0.0	85.5±0.0	93.5±0.0	99.7±0.0	**0.195±0.005**
	CatBoost [157]	**83.6±0.3**	**92.4±0.1**	87.2±0.2	92.8±0.1	77.5±0.0	85.8±0.0	96.4±0.0	99.8±0.0	0.196±0.004
	Model Trees [164]	82.6±0.2	91.5±0.0	85.0±0.2	90.4±0.1	69.8±0.0	76.7±0.0	-	-	0.385±0.019
Deep Learning	MLP [165]	73.2±0.3	80.3±0.1	84.8±0.1	90.3±0.2	77.1±0.0	85.6±0.0	91.0±0.4	76.1±3.0	0.263±0.008
	VIME [166]	72.7±0.0	79.2±0.0	84.8±0.2	90.5±0.2	76.9±0.2	85.5±0.1	90.9±0.1	82.9±0.7	0.275±0.007
	DeepFM [167]	73.6±0.2	80.4±0.1	86.1±0.2	91.7±0.1	76.9±0.0	83.4±0.0	-	-	0.260±0.006
	DeepGBM [168]	78.0±0.4	84.1±0.1	84.6±0.3	90.8±0.1	74.5±0.0	83.0±0.0	-	-	0.856±0.065
	NODE [169]	79.8±0.2	87.5±0.2	85.6±0.3	91.1±0.2	76.9±0.1	85.4±0.1	89.9±0.1	98.7±0.0	0.276±0.005
	NAM [170]	73.3±0.1	80.7±0.3	83.4±0.1	86.6±0.1	53.9±0.6	55.0±1.2	-	-	0.725±0.022
	Net-DNF [171]	82.6±0.4	91.5±0.2	85.7±0.2	91.3±0.1	76.6±0.1	85.1±0.1	94.2±0.1	99.1±0.0	-
	TabNet [158]	81.0±0.1	90.0±0.1	85.4±0.2	91.1±0.1	76.5±1.3	84.9±1.4	93.1±0.2	99.4±0.0	0.346±0.007
	TabTransformer [159]	73.3±0.1	80.1±0.2	85.2±0.2	90.6±0.2	73.8±0.0	81.9±0.0	76.5±0.3	72.9±2.3	0.451±0.014
	SAINT [160]	82.1±0.3	90.7±0.2	86.1±0.3	91.6±0.2	**79.8±0.0**	**88.3±0.0**	96.3±0.1	99.8±0.0	0.226±0.004
	RLN [172]	73.2±0.4	80.1±0.4	81.0±1.6	75.9±8.2	71.8±0.2	79.4±0.2	77.2±1.5	92.0±0.9	0.348±0.013
	STG [173]	73.1±0.1	80.0±0.1	85.4±0.1	90.9±0.1	73.9±0.1	81.9±0.1	81.8±0.3	96.2±0.0	0.285±0.006

Table 2.1: Open performance benchmark results based on (stratified) fivefold cross-validation on five real-world datasets Home Equity Line of Credit (HELOC) [174], Adult Income [77], Higgs bosons (HIGGS) [175], Covertype [77], and California Housing [176] from the "Deep Neural Networks and Tabular Data: A Survey" manuscript. We use the same fold splitting strategy for every data set. The top results for each data set are in **bold**, we also underline the second-best results. The mean and standard deviation values are reported for each baseline model. Missing results indicate that the corresponding model could not be applied to the task type (regression or multi-class classification).

and SAINT [160], claim that using attention maps one can get local explanations. In order to confirm this hypothesis, we examined the local feature attribution characteristics of these methods to the well-established KernelSHAP values in the community [161]. This selection of KernelSHAP is justified by the absence of ground truth attribution values.

Last but not least, we conclude our survey with an analysis of current trends and the identification of open research questions in the areas of self-supervised learning, transfer learning, tabular data generation, as well as other topics.

Tabular Data Open Performance Benchmarking

Despite recent advances in the research of deep neural networks for heterogeneous tabular data, to the best of our knowledge, there are no open-source benchmarks dedicated exclusively to tabular data. Consequently, the purpose of the work is to provide a fair and open-source benchmark for machine learning methods on tabular data that will help identify significant advances over the current state of the art.

To allow the results to be fully reproducible, we utilize the Docker container [177] and open-source Python packages. Furthermore, since the hyperparameter selection plays a huge role in the final performance of a machine learning model [178], we utilize a well-known optimization framework - Optuna [179]. We set the number of iterations to 100 for each model and cross-validate each set of hyperparameters using five folds. The hyperparameter ranges we use are available online along with our code. We carefully designed the search space based on the information provided in relevant papers and recommendations from the Optuna framework's authors. The full code is publicly available online on GitHub[1].

Table 2.1 presents results of our extensive evaluation using the proposed benchmark on five real-world tabular datasets. We used the five fold (stratified) cross-validation reporting the mean and standard deviation values. The details about datasets are in the our work in Appendix A.1.

Main findings of the "Deep Neural Networks and Tabular Data: A Survey" publication

We have identified the following as our main outcomes:

I The discussed in this thesis survey is the first to systematically examine the use of deep neural networks for heterogeneous tabular data. In this context, we summarize the main challenges and research advances in the modeling, generation, and explanation of tabular data using these techniques. We have collected and carefully reviewed over twenty-three methods for learning from tabular data using deep neural networks, as well as sixteen methods for generating tabular data using these techniques. Our survey provides a comprehensive overview of the current state of the field and identifies areas for further research and development.

II We introduced a unified taxonomy for deep neural network-based methods for heterogeneous tabular data (Fig. 2.1), categorizing the existing approaches into three main groups and found subgroups.

III We also discussed and assessed explanation properties of the existing deep learning methods for tabular data. Based on our experiments, we showed that the TabNet model [158] provides more close to a target explanations, in comparison to other baselines.

IV We developed an open-source and fully-reproducible benchmark for fair machine learning models comparison on tabular data. Utilizing our open performance benchmark, we conducted an unbiased evaluation of the twenty machine learning models on five real-world datasets, with various number of samples from 10,000 to 10,000,000, and from different domains. In order to compare the machine and

[1]Open performance benchmark on tabular data: `https://github.com/kathrinse/TabSurvey`

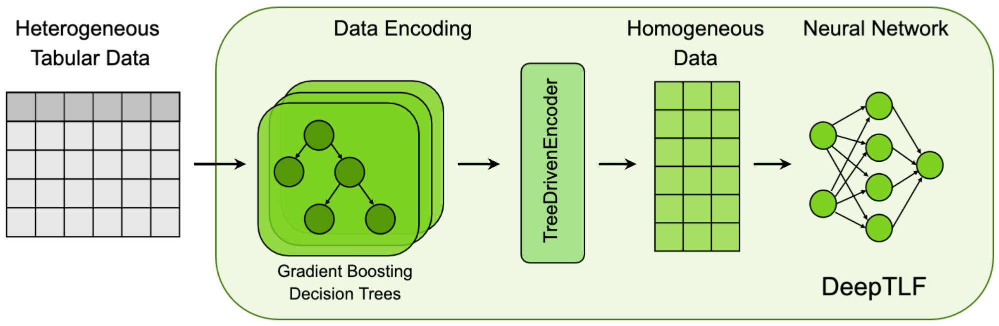


Figure 2.2: A data pipeline for the proposed DeepTLF framework. First, the *train* data is used to train a gradient boosted decision trees model. The heterogeneous data is transformed by exploiting the structures of the decision trees in the ensemble. More specifically, the *TreeDrivenEncoder* algorithm distills information from trained decision trees of the gradient boosting decision tree model to produce homogeneous binary vectors. These vectors are then used to train a deep neural network. Even considering the construction of the gradient boosted decision tree model as preprocessing for DeepTLF, the overall preprocessing time is lower than that of typical data preprocessing for deep neural networks, such as normalization or handling missing or categorical values.

deep learning methods fairly, we tune hyper-parameters for each model for hundred steps. Furthermore, we utilize the (stratified) fivefold cross-validation scheme for obtained final results.

V Based on the empirical evaluation, we observed that deep neural network-based methods for heterogeneous tabular data are still, in general, inferior to machine learning methods based on gradient-boosting decision tree ensembles such XGBoost [155], LightGBM [156], or CatBoost [157]. Only on very large data set with eleven million rows, a Transformer-based model SAINT [160] showed better performance then the rest of baseline models.

VI We conclude the presented manuscript with a discussion on current trends for the deep learning on tabular data. Additionally, we have identified a number of open research questions and challenges that require further exploration.

All our results have important implications for further research in area of learning on heterogeneous tabular data, especially in the context of deep neural networks. To the best of our knowledge, the discussed manuscript is the first comprehensive overview of deep learning techniques for heterogeneous tabular data. As such, this work can serve as a useful guide for researchers and practitioners working on data-driven tasks with tabular data. In the following Subsection, a novel deep tabular learning model is presented.

2.1.2 DeepTLF: A Novel Deep Tabular Learning Framework

This Subsection is based on the journal publication *"DeepTLF: Robust deep neural networks for heterogeneous tabular data."* For a complete text version of the manuscript, please refer to the Appendix A.1.

Prior to this, we discuss that deep neural networks constitute the state of the art on the mostly homogeneous data, e.g., visual, textual, or sound datasets, where tabular data is often highly heterogeneous [11, 12, 1] (Sec. 1.3). Also, as we empirically demonstrate it in our survey, where even the specialized for tabular data deep models cannot outperform the properly-tuned gradient boosting decision tree models (Sec. 2.1.1, Appendix A.1).

The primary motivation for this work is to propose a novel robust deep learning approach for heterogeneous tabular data, as a response to the mediocre results that deep neural networks have achieved on the tabular data modality. In order to accomplish the objective and overcome the data heterogeneity issue, we provide a novel deep tabular learning framework, coined DeepTLF. As discussed in Chapter 1, the challenge stems from the concurrent existence of numerical and categorical feature types, complex, irregular dependencies between the features, and other data-related issues such as scales, outliers, and missing values [1]. In contrast to deep learning-based methods, decision tree ensemble methods require minimal data preprocessing and are robust to inappropriate training data [180, 181].

The proposed DeepTLF approach is a hybrid machine learning model that combines the data preprocessing power of decision tree-based ensemble algorithms with the inherent flexibility of deep neural networks. Figure 2.2 presents the DeepTLF data pipeline, it consists of a heterogeneous tabular dataset, data encoding block, and deep neural network. Considering that decision-tree-ensemble-based models are able to provide robust data preprocessing powers, such as eliminating the need for scaling variables or filling in the missing values, which is one of the main issues for deep models on tabular data - information loss due to the data preprocessing [1].

For the data transformation step in the encoding block we propose a novel knowledge distillation approach - TreeDrivenEncoder. It utilizes the structure of trained decision trees from the state-of-the-art tabular data model gradient boosting [1]. Essentially, TreeDrivenEncoder distills knowledge from a gradient boosting model by exploiting Boolean expressions present in decision nodes. The purpose of this stage can be described as "feature engineering," which aims to produce a more homogeneous data representation that is more suitable for usage with deep neural networks. Furthermore, According to He et al. [182], the boosted decision tree transformation can be seen as a supervised feature encoding that transforms the original feature space into a more condensed binary-valued feature space. This is because each tree is designed to model the residual of the previous tree in each iteration.

In the literature, numerous approaches have been developed to distill knowledge from the leaf outputs of random forest or gradient boosting decision tree algorithms, which have been extensively examined [183, 184, 182, 168, 185]. These methods demonstrate improvements in a variety of applications, such as online advertising [186] and recommendation system [187]. As opposed to previously published state-of-the-art approaches, the DeepTLF method utilizes the entire decision tree structure from a gradient boosting decision tree model and considers each feature's representation independently when distilling information; thus, it does non-linearly transformation of tabular data in the data encoding step.

By combining gradient-boosted trees, which have the ability to handle various scales, attribute types, missing values, and outliers, with neural networks, we are able to achieve excellent predictions. On average, the DeepTLF approach demonstrates a 19.6% improvement in performance across several tabular datasets compared to standard deep learning models.

In addition, we also address the issue of multimodal learning, which involves a combination of various data modalities, namely visual and tabular data [188, 189, 190, 191, 39, 192, 193, 194]. Currently, deep learning-based models represent the state-of-the-art in multimodal learning [195], even when heterogeneous tabular data is involved. This is due to the challenge posed by applying decision tree ensemble methods in the presence of other data modalities. With the approach presented in this paper, multimodal problems involving tabular data can be addressed in an integrated manner, while also achieving robust performance. Notably, our approach to a multimodal strategy does not rely on any specific artificial neural network architecture, thereby enabling easy integration into existing multimodal data workflows.

The implementation of TreeDrivenEncoder and DeepTLF methods is open-sourced and published on GitHub[2]. For qualitative and quantitative results of the proposed DeepTLF method, please refer to Appendix A.1.

Main findings of the "DeepTLF: Robust Deep Learning on Tabular Data" publication

Our work has resulted in the following main outcomes:

I We demonstrate that one of the major challenges for the deep neural networks on tabular data is its heterogeneity. To accomplish that, we selected seven real-world tabular datasets with the sample size from 19,000 to 11,000,000 samples, and compare the DeepTLF method to eleven baselines on six challenging experiments. Across all datasets and experiments, the proposed framework shows high results.

II Furthermore, we present a novel scheme for encoding tabular data called TreeDrivenEncoder, which does the non-linearly transformation from heterogeneous tabular data into homogeneous data modality.

[2]DeepTLF framework: `https://github.com/unnir/DeepTLF`

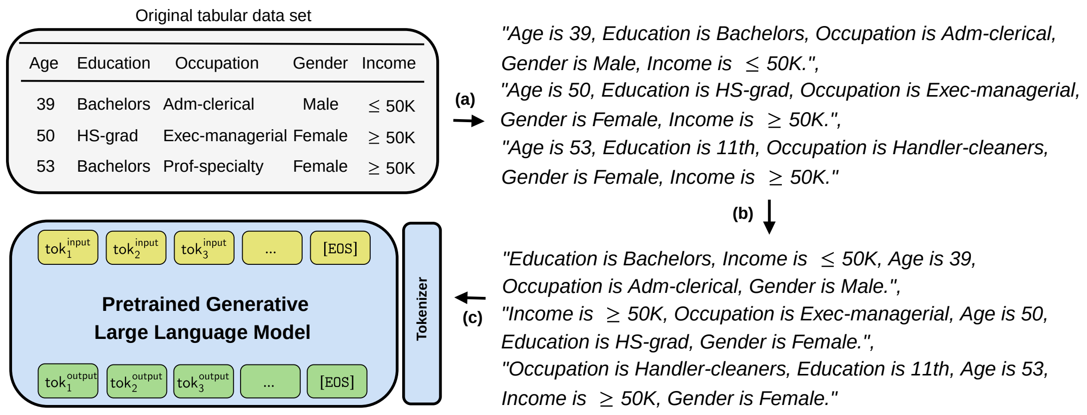

Original tabular data set
Age Education Occupation Gender Income
39 Bachelors Adm-clerical Male ≤ 50K
50 HS-grad Exec-managerial Female ≥ 50K
53 Bachelors Prof-specialty Female ≥ 50K
(a)
"Age is 39, Education is Bachelors, Occupation is Adm-clerical, Gender is Male, Income is ≤ 50K.",
"Age is 50, Education is HS-grad, Occupation is Exec-managerial, Gender is Female, Income is ≥ 50K.",
"Age is 53, Education is 11th, Occupation is Handler-cleaners, Gender is Female, Income is ≥ 50K."
(b)
"Education is Bachelors, Income is ≤ 50K, Age is 39, Occupation is Adm-clerical, Gender is Male.",
"Income is ≥ 50K, Occupation is Exec-managerial, Age is 50, Education is HS-grad, Gender is Female.",
"Occupation is Handler-cleaners, Education is 11th, Age is 53, Income is ≥ 50K, Gender is Female."
(c)
Tokenizer
tok_1^{input} tok_2^{input} tok_3^{input} ... [EOS]
Pretrained Generative Large Language Model
tok_1^{output} tok_2^{output} tok_3^{output} ... [EOS]

Figure 2.3: The GReaT data pipeline for the fine-tuning step. First, a textual encoding step transforms tabular data into meaningful text (a). Subsequently, a feature order permutation step is applied (b), before the obtained sentences can be used for the fine-tuning of a large language model (c). The toy tabular data set inspired by the Adult Income data set [77].

III Within the DeepTLF framework, we also address the issue of the multimodal learning. And empirically demonstrate that our approach is superior to the common data fusion technique, relying on the deep neural networks.

As a final note in this Section, results from both works have substantial implications for further research in the area of learning on heterogeneous tabular data, especially in the context of deep neural networks.

2.2 Realistic Tabular Generation using Transformer-networks

This Section is based on a single publication *"Language Models are Realistic Tabular Data Generators"*, the paper is accepted to the The Eleventh International Conference on Learning Representations (ICLR) 2023. For a complete text version of the manuscript, please refer to the Appendix A.1.

GReaT: Large Language Models are Realistic Tabular Data Generators

In line with what we discussed previously, tabular data is one of the most common forms of data in the machine learning field. There is a wide array of real-world applications that

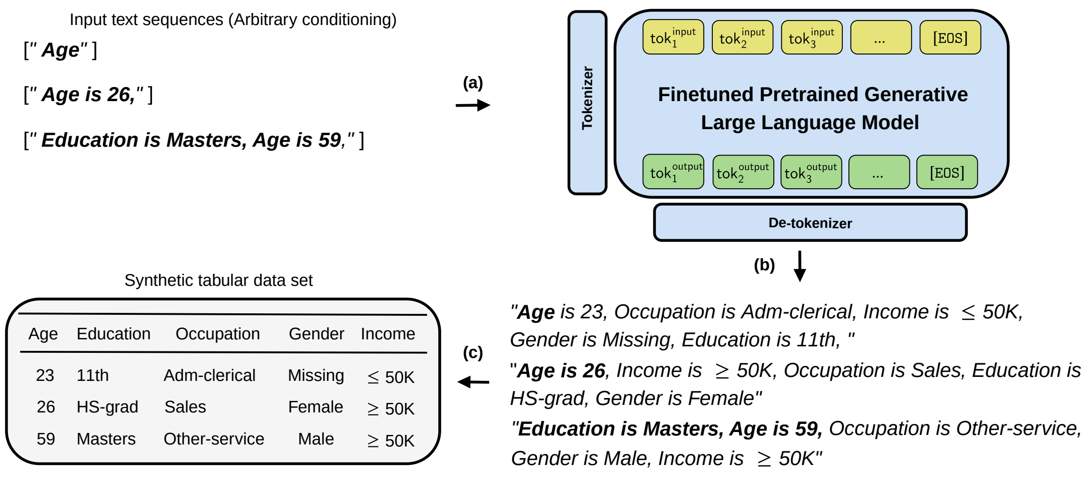

Input text sequences (Arbitrary conditioning)
["Age"]
["Age is 26,"]
["Education is Masters, Age is 59,"]
(a)
Tokenizer
Finetuned Pretrained Generative Large Language Model
[EOS]
De-tokenizer
(b)
"Age is 23, Occupation is Adm-clerical, Income is ≤ 50K, Gender is Missing, Education is 11th, "
"Age is 26, Income is ≥ 50K, Occupation is Sales, Education is HS-grad, Gender is Female"
"Education is Masters, Age is 59, Occupation is Other-service, Gender is Male, Income is ≥ 50K"
(c)
Synthetic tabular data set
Age Education Occupation Gender Income
23 11th Adm-clerical Missing ≤ 50K
26 HS-grad Sales Female ≥ 50K
59 Masters Other-service Male ≥ 50K

Figure 2.4: The sampling procedure of the proposed method for synthetic data generation. In order to generate new data points using a pretrained large language model, it is necessary to include either a single feature name or an arbitrary combination of feature-value pairs into text (a). Subsequently, the input sequence is completed by the fine-tuned large language model (b) and can be transformed back into a tabular format (c). In comparison to other state-of-the-art approaches, GReaT allows arbitrary conditioning on feature subsets without model retraining, i.e., the sampling can be performed by conditioning on any feature name or combination of feature names and values.

rely substantially on tabular data. However, as a result of the high cost of data collection, heterogeneous tabular data sets tend to be class imbalanced.

For example, tabular data sets tend to have long-tail label distributions. Furthermore, there are a number of reasons that may prevent the sharing of critical person-related information, including the impurity issues that often impede the application of modern machine learning algorithms, such as noisy or missing values. In order to alleviate these crucial issues, synthetic tabular data is used.

A recent research for synthetic tabular data generation mostly based on generative deep learning architectures [1], for example variational autoencoders or generative adversarial networks, are widely adopted for heterogeneous tabular data generation due to their success in other homogeneous data modalities.

However, as we have already observed to this point, heterogeneous tabular data sets represent a significant challenge for deep neural network models because they require multiple preprocessing steps and the modeling of structured data with a variety of variable types and distribution forms [1]. With a view to resolving the issue, several popular generative models from the computers vision domain have been adapted for tabular data – variational autoencoders or generative adversarial networks, for example CTGAN [102], TVAE [102], and so forth [1].

As we pointed in our work [6], the heterogeneity of feature types and values leads to three core challenges in tabular data preprocessing and modeling:

- **Extensive and lossy preprocessing**. Many of the existing methods for generating tabular data require significant preprocessing, including converting categorical data into numerical values, scaling or normalizing the data, replacing missing values, and removing outliers and smoothing the data. All this may result in information loss, and strongly influence the quality of synthetic samples.

- **Context knowledge for coherent semantics**. As we pointed in Sec. 1.4, often variables in a tabular dataset have clear unambiguous relationship. Therefore, it is necessary to learn this connects for a realistic synthetic data generation task.

- **Arbitrary conditioning**. A model that is versatile and can generate data for various applications should be able to synthesize data based on an arbitrary set of variables. This allows for the imputation of missing data patterns or oversampling of specific subsets. Currently, not all of the modern and state-of-the-art tabular data generation models provide arbitrary conditioning. For example, the CTGAN [102] allows conditioning only on a single categorical value.

There is limited research on the use of deep autoregressive large language models in heterogeneous tabular data, which are generative in nature, for example generative pretrained transformers (GPT) models [197, 198, 199]. Additionally, Transformers have demonstrated superior performance in many generation tasks. These findings suggest that the use of Transformers in the context of heterogeneous tabular data could be a promising area of study.

Therefore the main motivation of our work is to present a novel approach for heterogeneous tabular data generation using pretrained large language models, provide support for the arbitrary conditioning power, and last but not least, improve the quality of the synthetic data.

To maximize the usage of pretrained large language models, we propose a novel *textual subject-predicate-attribute encoding scheme* for tabular data. This scheme transforms a sample (row) into a set of clauses that form a sentence, utilizing the contextual information provided by the variable names. To put it simply, this means given a tabular dataset with variable names, we can construct a sentence by concatenating clauses of a feature name and value for each row. Fig. 2.3 (a) illustrates the process using a toy example. Then in order to achieve the fully-arbitrary sampling procedure we utilize two components: the autoregressive large language model and random permutation step during the fine-tuning stage (Fig. 2.3 (b)). The sampling procedure is depicted in Fig. 2.4.

Figure 2.5 demonstrates the comparison of the original data and synthetic data using two variables from the California housing dataset [176]. It is evident that the proposed model synthesizes the most plausible tabular data samples.

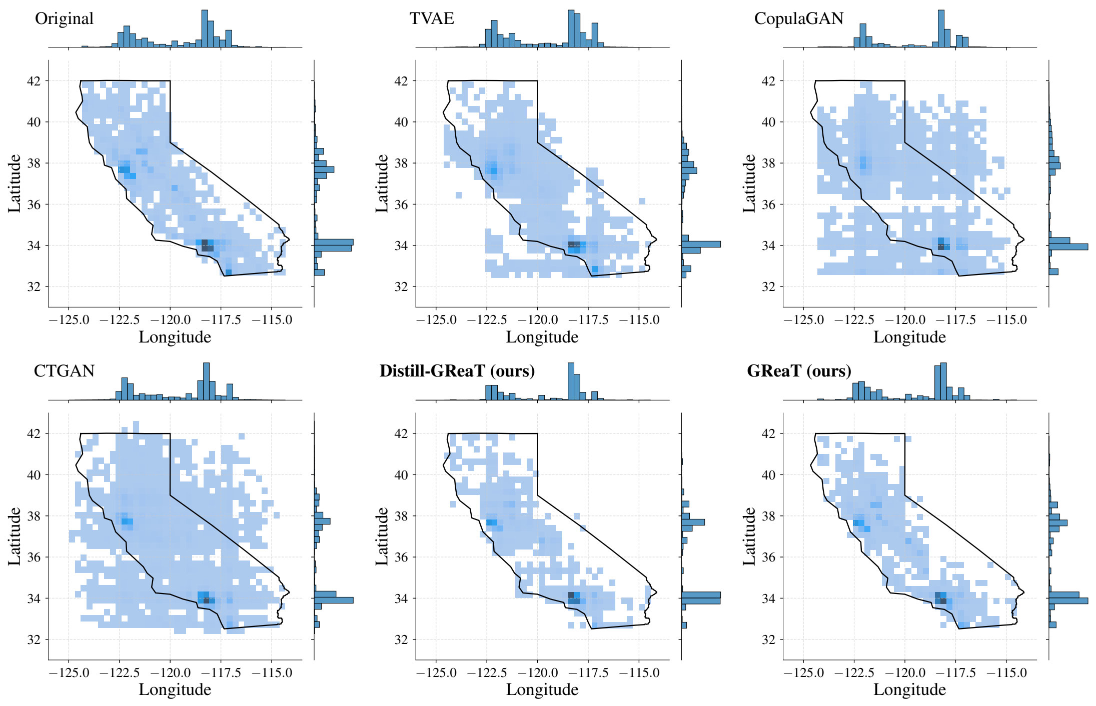


Figure 2.5: A comparison of the original and generated samples for the California Housing data set [176], which contains characteristic information about different properties in California, USA. We show joint histogram plots of the highly interconnected variables `Latitude` and `Longitude`. The black outline indicates the true boundary of the state of California. We select the following baselines for the comparison TVAE [102], CopulaGAN [196], and CTGAN [102].

We developed and published online an easy-to-use Python framework[3], which requires only a few lines of code to get the synthetic data (Fig. 2.6).

Main findings of the "Language Models are Realistic Tabular Data Generators" publication

We have identified the following as our main outcomes of the work:

I We demonstrate that the large language models are realistic tabular data generators though an extensive set of challenging experiments. In comparison to GAN/VAE-based method, our GReaT approach shows an average improvement of 20% across on six real-world datasets on the discriminator metric (Sec. 1.4.2).

[3]Open-source implementation of the GReaT framework: `https://github.com/kathrinse/be_great`

```python
# pip install be-great
from be_great import GReaT
from sklearn.datasets import fetch_california_housing

data = fetch_california_housing(as_frame=True).frame

model = GReaT(llm='distilgpt2', batch_size=64, epochs=50)
model.fit(data)
synthetic_data = model.sample(n_samples=100)
```

Figure 2.6: A Python code example of the GReaT framework for the California housing tabular dataset [176] from an open-source Scikit-Learn package [200]. This example illustrates the simplicity of using the GReaT framework to model tabular data.

II We present a novel textual encoding technique for tabular data, which allows the usage of the pre-trained models from the different data modality. Thereby, we connect tabular and textual data modalities via a textual encoding scheme.

III Our proposed GReaT demonstrates that autoregressive models are effective at generating samples, allowing us to achieve the ability to condition generation on any combination of variables, regardless of their type. This capability has potential applications in a variety of contexts involving tabular data, such as generating more comprehensive synthetic data, imputing missing values, and addressing class imbalance issues.

To conclude, we believe that our research involving the application of large language pre-trained models on heterogeneous tabular data represents a step closer to new possibilities for the generation of heterogeneous data in the future.

2.3 Robust Explainability of Deep Neural Networks

This section is based on four manuscripts: *"CancelOut: A Layer for Feature Selection in Deep Neural Networks"* published at the international conference on artificial neural networks (ICANN), 2019, and *"A Robust Unsupervised Ensemble of Feature-Based Explanations using Restricted Boltzmann Machines"* published in the 1st Workshop on eXplainable AI approaches for debugging and diagnosis at the conference on Neural Information Processing Systems (NeurlPS), 2021, *"Consistent and Efficient Evaluation Strategy for Attribution Methods"* published at the international conference on machine learning (ICML), 2022, and *"Relational Local Explanations"*, which is currently under submission to an international conference.

The proposed approaches are not only applicable to tabular data, but also to visual and textual data modalities, which are common data formats in the field of machine learning. The presented approaches demonstrate the versatility and potential for broad application in various data analysis and modeling tasks.

The Section is organized in the following manner: it consisted of four Subsection for each work, where motivation and main findings are provided.

2.3.1 Relational Local Explanations

Most current methods for post-hoc explanation in machine learning models generate individual feature attribution scores for each variable, overlooking an essential aspect such as the interplay among features, which is particularly significant in visual and text data. In order to address this, we introduce a unique algorithm for feature attribution that is not only model-agnostic but also permutation-based, focusing on the relationships between input variables. This approach provides a more comprehensive insight into machine learning models and data decisions. The local explanations produced through this method directly assess interactions between local features, covering another critical dimension of explanations. Our experimental assessments using both image and text data types substantiate the effectiveness of our framework and the credibility of the explanations generated.

We present a model-agnostic approach to local explanations, termed Relative Local Explanations (RLE). This approach addresses one of the challenges of post hoc explanations - interpretability of inter-variable relationships by providing a qualitative measure of how feature attributions interact. The formal definition of is listed below:

Definition 5 (Relational Local Explanation) *A relational local explanation function can be seen as $\Psi(f, \boldsymbol{x}, c_x) \in \mathbb{R}^{n \times n}$, where $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is a black-box model as above (Sec. 1.5) and $\boldsymbol{x} \in \mathbb{R}^n$ is an input sample belonging to a class $c_{\boldsymbol{x}} \subset \mathbb{R}$. The output of the Ψ is a relational explanation representation in a form of a adjacency matrix $\mathbf{A}_{\boldsymbol{x}}$.*

This form of explanation is helpful for various data types and problems where information about the relationship is also essential. Furthermore, the RLE framework provides standard local explanations in the form of feature attributions. As a final point, the proposed feature attribution method, RLE, has shown excellent results in visual and quantitative experiments when compared with state-of-the-art approaches in the field.

We have made the implementation of our RLE framework publicly available by publishing and open-sourcing it.[4] This allows for greater transparency and accessibility for interested researchers and practitioners in the field. By open-sourcing the implementation, we hope to facilitate further exploration and advancement of the RLE framework within the academic community.

Main findings of "Relational Local Explanations"

We have identified the following as our main outcomes of the work:

I Our study emphasizes the importance of relational interactions among input features in explaining local patterns for more holistic local explanations.

II In light of the fact that visual and textual data are compositional by nature, i.e. there exists "regional" information between variables, it is imperative to not only identify and quantify the variables that are most important in a given data sample, but also to highlight the combinations of variables that are most critical.

III We developed and formally described a model-agnostic local feature attribution technique, coined relational local explanations. To the best of our knowledge, our proposed explanation framework is the first model-agnostic local explanation algorithm based on the relationships between input variables.

2.3.2 A Robust Unsupervised Aggregation of Local Explanations Using Restricted Boltzmann Machines

One of the major challenges in the field of machine and deep learning is the ability to provide clear and consistent (local) explanations for the decisions made by these models. In recent years, many approaches have been developed to address this issue, but one of the key problems with the local feature attribution methods is that they often produce divergent explanations for the same machine learning model [4], please refer to Table 2.2 for visual comparison of the local explanations. This can make it difficult to understand how and why a model is making a particular decision, and it can also lead to confusion and inconsistency in real-world applications.

Furthermore, the same local explanation algorithm can produce varying saliency maps depending on the chosen hyperparameters or even the random seed used [206]. We refer to this frequently occurring phenomenon as *local-explanation discrepancy*. In addition to this, it is worth noting that obtaining the ground truth for a local explanation is not

[4]An open-source implementation of the RLE framework: `https://github.com/unnir/rle`

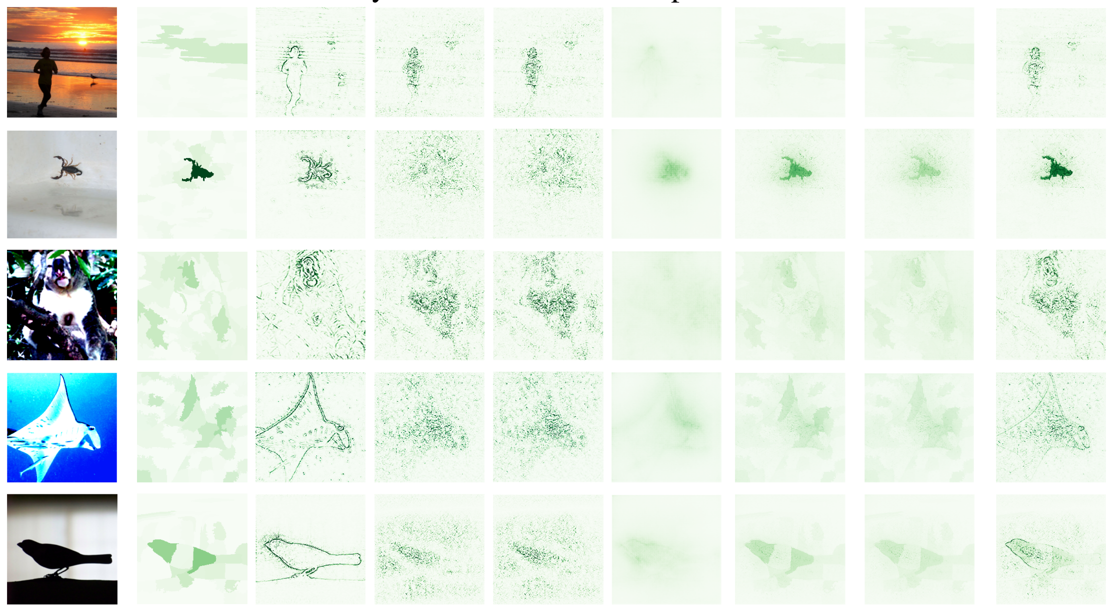
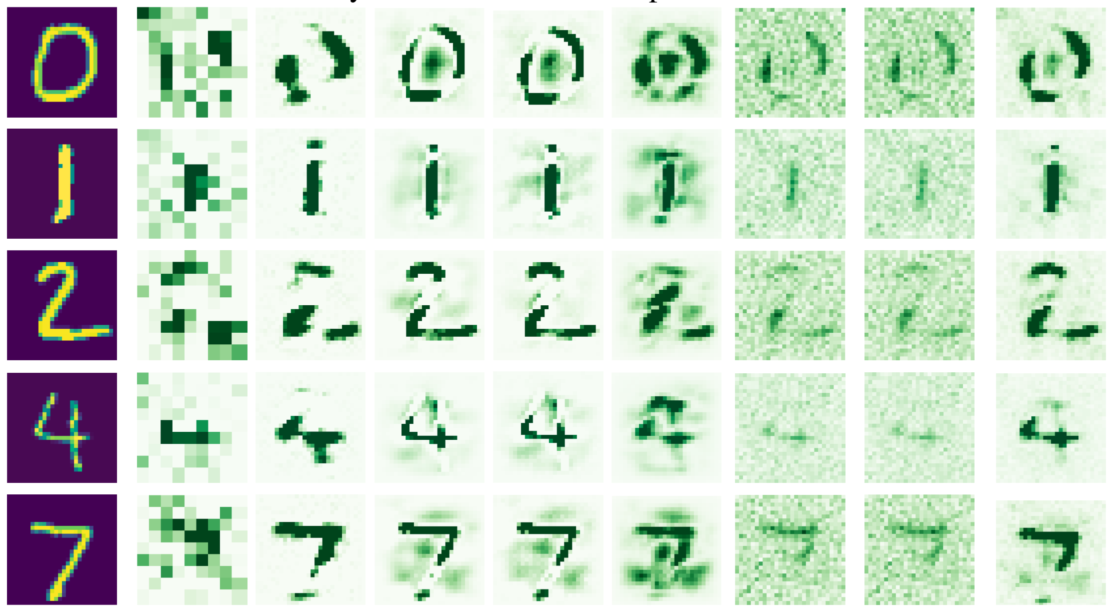
Original	LIME [118]	GB [202]	IG [201] [119]	GS [203]	SG	Mean ensemble	Variance ensemble	**RBM ensemble**
Without noisy feature attribution maps in the ensemble								
								
With noisy feature attribution maps in the ensemble								
								

Table 2.2: A visual comparison between baseline methods and ensemble methods on image data from the ImageNet [204] and MNIST [205] datasets.

currently possible [49, 4]. Moreover, the complexity of the model and the data used for training can also impact the accuracy and reliability of the local explanation. As a result, it is important to carefully evaluate and compare different local explanation methods in order to choose the most appropriate one for a given task.

Based on aforementioned considerations, the main motivation behind this work can be summed up as follows: (i) First, verify the local-explanations discrepancy of state-of-the-art explanation methods. (ii) Next, provide a novel solution for the effect minimization of the local-expiation discrepancy and theoretically define it. (iii) Lastly, derive a probabilistic aggregation of existing techniques for feature-based, local explanations and a corresponding easy-to-use Python framework for robust local explanations.

In the machine learning community, ensemble learning is often utilized to amalgamate methods that may not consistently agree with one another [207, 208]. Compared to non-ensemble methods, there is generally a marked improvement in performance, with increased robustness to outliers and noisy data [209, 210]. The principle behind ensemble learning is the combination of multiple machine learning methodologies to enhance the overall system performance and mitigate the effects of individual errors by each method. This approach, backed by both statistical learning theory and practical applications, is considered a more robust strategy for constructing machine learning systems [211, 212].

Due to the lack of ground truth, we propose using restricted Boltzmann machines [213] for unsupervised ensemble learning in order to generate reliable and robust feature-based explanations for deep neural networks. This approach builds upon existing work on unsupervised ensembling learning [214, 207] by aggregating the results of different feature-based explanation methods in a probabilistic manner. Furthermore, previous research has demonstrated the effectiveness of restricted Boltzmann machines in a truth discovery setting, which is similar to our task of finding an accurate feature importance map from multiple sources [215].

In 2016, Shaham et al. published a paper [214] demonstrating that the Dawid and Skene model [216], which presupposes independence among models, is equivalent to a restricted Boltzmann machine with a single hidden node. Similar probabilistic approaches have also been employed by Kasneci et al. [217, 218] to aggregate information from multiple independent sources. Therefore, drawing insights from prior work, we have developed a technique to unearth latent explanations using restricted Boltzmann machines. In our research, we operate under the assumption that the true local explanation can be approximated using a restricted Boltzmann machine with one hidden unit, representing whether an input feature or pixel is relevant for the final explanation or not.

To validate our approach, we employ several state-of-the-art feature attribution methods, to generate local explanations for deep neural network models. These explanations provide insight into the factors that influenced the model's prediction for a specific input. Next, we aggregate the local explanations from the ensemble of models to generate a single "true feature attribution", i.e., reliable explanation, through the use of restricted Boltzmann machines.

By combining the local explanations from multiple models, our proposed method can provide a more accurate and robust interpretation of the underlying data and prediction. This can be useful for a variety of applications, such as improving the transparency and trustworthiness of machine learning models, or for identifying potential biases or weaknesses in the models.

We demonstrated the robustness and reliability of the proposed approach through qualitative and quantitative experiments. With noisy attribution maps in an ensemble, the proposed approach successfully selects only valuable information, mitigating irrelevant local importance (please refer to Fig. 2.2). Furthermore, our work clarifies and mitigates the problem of contradictory results that might be obtained using different explanation and evaluation methods. Lastly, our approach can also be used within a single interpretability framework to reduce the sensitivity to hyperparameters of a feature-based explanatory approach.

The corresponding implemented framework for the robust aggregation of local explanation is open-sourced and available online.[5]

Main findings of the "A Robust Unsupervised Aggregation of Local Explanations Using Restricted Boltzmann Machines" work

We have identified the following main outcomes from this work:

I We demonstrate and theoretically justify that the aggregation of feature attributions using restricted Boltzmann machines leads to more robust local explanations.

II The proposed aggregation method can be applied to a single model or multiple models in an ensemble.

III Our approach has several benefits, in addition to providing interpretability, it can be used for debugging and diagnostic purposes. This can help increase the long-term trust in and acceptance of deep learning in real-world applications.

2.3.3 A Consistent and Efficient Evaluation of Local Explanations

A variety of local feature attribution methods have been proposed in recent years [44, 45, 46, 47, 49], and follow-up research has suggested a number of strategies for evaluating these methods [149, 150, 151, 219]. An evaluation strategy that is becoming increasingly popular is feature perturbation. It is used to determine attribution quality across different attribution techniques. Nevertheless, research has found that different evaluation strategies produce conflicting rankings of attribution methods and that computing these rankings can be prohibitively expensive [219, 5]. The main motivation of this study is the identification of bias in the pixel perturbation-based evaluation strategies, which causes inconsistent results. In order to accomplish the objectives, we present an analysis based

[5]Robust local explanations using RBMs framework
`https://github.com/JohanvandenHeuvel/AggregationOfLocalExplanations`

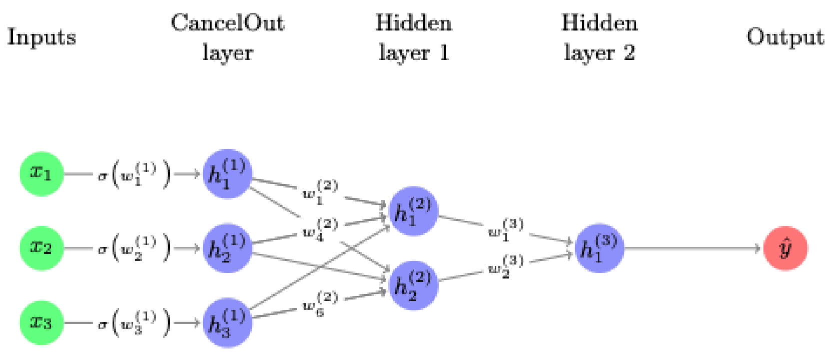


Figure 2.7: A deep neural network with the CancelOut layer as an input layer, where x_1, x_2, x_3 are input variables, and $\hat{y}$ is estimated output.

on information-theoretic principles of pixel perturbation-based evaluation strategies and present a novel algorithm which can be utilized for the comparison of non-contradictory feature attributes. The code is open-sourced and publicly available.[6]

Main findings of the "A Consistent and Efficient Evaluation of Local Explanations" work

We have identified the following as our main outcomes of the work:

I By conducting a comprehensive analysis based on information theory, we examined the underlying principles of perturbation-based evaluation techniques and found that the outcomes can be significantly distorted. To address this issue, we propose and demonstrate the effectiveness of the Noisy Linear Imputation approach, which mitigates bias. Our method also leads to a significant reduction in hyperparameters, such as the order of removal.

II Our research introduces a new evaluation framework called Remove and Debias (ROAD), which offers two benefits. Firstly, it reduces the impact of bias, leading to improved consistency among evaluation techniques. Secondly, ROAD does not require the computationally expensive step of retraining, resulting in a potential cost saving of up to 99 % compared to Remove and Retrain (ROAR) [219].

2.3.4 Global Explanations Through the CancelOut Layer

Another crucial aspect of the deep learning explanations is to obtain the global variable importance for a given black-box model. In this work, we propose a new layer for deep neural networks, coined CancelOut, which can be utilized for global feature attribution

[6]Remove And Debias (ROAD) framework
`https://github.com/tleemann/road_evaluation`

ranking in the context of both supervised and unsurprising learning. Additionally, the CancelOut layer can be used for (i) feature selection tasks, which in turn can be used to (ii) minimize the unwanted effects associated with the curse of dimensionality problem and to minimize (iii) training time by selecting the most informative variables; (iv) last but not least, it helps to improve the predictive performance and robustness of a model as well.

An intuitive explanation for this is that it is necessary to update the weights of CancelOut during a training stage, so that “noisy” or less informative features will be canceled out with a negative weight. In such a case, the variables that contribute most to the learning process will be given a positive weight. The CancelOut input can be viewed as a “gate” input; a deep neural network determines which information (related to features) should pass through. A simple deep learning model with the CancelOut layer depicted in Fig. 2.7.

As a formal definition of the CancelOut layer, we have the following:

$$CancelOut(\boldsymbol{x}) = \boldsymbol{x} \odot g(\boldsymbol{w_{co}}), \tag{2.1}$$

where $\odot$ indicates an element-wise multiplication, $\boldsymbol{x}$ is an input vector $\boldsymbol{x} \in \mathbb{R}^N$, $\boldsymbol{w_{co}}$ is a weight vector $w_{co} \in \mathbb{R}^N$, N is the feature size, and g is an activation function. Note $g(\boldsymbol{x})$ denotes here element-wise application, e.g. for a two-dimensional vector $\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$, then $g(\boldsymbol{x}) = g\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) = \left(\begin{bmatrix} g(x_1) \\ g(x_2) \end{bmatrix}\right)$.

The CancelOut layer is differentiable in its entirety and can be integrated with various activation functions. The weight parameter $\boldsymbol{w_{co}}$ is learned during the training process using the backpropagation algorithm [220].

In addition, we propose a theorem that theoretically demonstrates the effectiveness of the CancelOut layer in identifying the most influential variables based on a given loss function. The formal proof of this theorem can be found in the appendix of the accompanying paper (Appendix A.2).

In terms of functionality, the CancelOut layer provides a very simple user interface that makes it very easy to configure and use. To add this to a deep learning training loop, only a few lines of code need to be added to the code. We have published and open-sourced the CancelOut implementation[7] for major deep learning frameworks such as PyTorch [221] and TensorFlow/Keras [222, 223].

Main findings of the “CancelOut: A layer for feature selection in deep neural networks” work

We have identified the following as our main outcomes of the work:

[7]CancelOut layer open-source implementation: `https://github.com/unnir/CancelOut`

I We introduced a novel feature ranking method for deep neural networks, which can be used in supervised settings for classification or regression tasks. In addition, the CancelOut layer is suitable for unsupervised tasks that are conducted based on the deep autoencoder architecture.

II We demonstrate that deep learning models are capable of self-feature selection using the CancelOut layer.

III Due to the power of the deep neural networks, the presented approach learns linear and non-linear data dependencies, which helps to identify the complex interaction between features for the global explanation scoring. Thus, the proposed layer helps understand the data and its influence on the performance of deep learning models.

2.4 A Summary of the Discussed Publications

In this section, we provide a brief overview of the main motivations and significant findings that led to the development of the work discussed here. There is a wide range of contributions made by the dissertation. First, we present an extensive and thorough overview of existing deep learning methods for heterogeneous tabular data inferencing and generation tasks. We also developed a novel hybrid algorithm called DeepTLF, which combines the preprocessing power of decision tree ensemble algorithms and the flexibility of deep neural networks.

In the context of synthetic tabular data generation, we present a novel approach called Generation of Realistic Tabular data (GReaT) that utilizes autoregressive pretrained large language models. Our work is particularly noteworthy for eliminating the process of data preparation and simultaneously modeling all variables, thereby eliminating the need for data transformation or missing values imputation steps. A series of experiments were carried out to quantify the validity and quality of the produced data samples from multiple angles, demonstrating the effectiveness of the proposed approach. Our results demonstrate that GReaT maintains state-of-the-art performance across a wide range of real-world data sets.

Furthermore, in this dissertation, we have approached the topic of explainable machine learning from a variety of perspectives. Specifically, we address the issue of hyper-parameter sensitivity and propose a solution based on restricted Boltzmann machines for the problem. We also present a novel approach that takes the spatial correlation between input variables and generates relational local explanations for them. Our last, but not least, contribution is the presentation of a method for evaluating local explanations with a highly-efficient approach.

Each of the papers discussed in this chapter is fully presented in Appendix A. Additionally, we have open-sourced our algorithm implementations and made them available online.

Chapter 3

Discussion

This chapter presents a discussion of the manuscripts summarized in Chapter 2 and fully presented in Appendix A, addressing three main topics: deep neural networks for heterogeneous tabular data, synthetic tabular data generation, and explainability of deep models. Toward the end of the Chapter, an outlook of future developments is presented.

3.1 Deep Tabular Learning

As part of this dissertation, we provide an extensive summary of the state-of-the-art algorithms for deep tabular learning on tabular data for inference and generation tasks by categorizing them into three groups, establishing a unified taxonomy of the approaches (Fig. 2.1). Furthermore, we propose an open-source benchmark for machine learning on heterogeneous tabular data, which is publicly available. Using our evaluation framework, we evaluate over twenty machine and deep learning models on five real-world tabular datasets from various domains and sizes. According to our results (Tab. 2.1.1), gradient-boosted tree ensemble algorithms continue to outperform deep learning models on supervised learning tasks, suggesting that progress in developing competitive deep learning models for tabular data has slowed down.

Moreover, we have proposed a novel algorithm for tabular data – DeepTLF, which is a combination of decision trees and deep neural networks (Sec. 2.1.2). The method can also be used for problems where multiple modalities exist along with tabular data.

Below, we discuss the current trends and open issues in the deep tabular learning domain. In addition to this, we provide a discussion on the proposed DeepTLF model.

Unified benchmarking for tabular data. The machine-learning community does not have an agreement on how to make a fair and efficient comparison of statistical models with tabular data. For example, Kadra et al. [83] uses about fourteen different tabular data sets for assessing the predictive performance of machine learning algorithms, Shwartz-Ziv & Armon [25] show that the choice of benchmarking data sets can have a non-negligible impact on the performance assessment. While in the presented dissertation in Sec 2.1.1, we chose common data sets with varying characteristics for the experiments, different choices of data sets, hyperparameters, or preprocessing decisions such as the encoding used for categorical data (e.g., one-hot encoding or other binary encoding schemes) may lead to different outcomes. Because of the excessive number of tabular datasets, there is a necessity for a standardized benchmarking procedure, which allows to identify significant progress with respect to the state-of-the-art models. Toward this goal, as a part of this dissertation, an open-source benchmark for deep learning models on tabular data is introduced 2.1.1. We strongly believe that it is critical to share reproducible results, therefore we suggest a usage of a Docker container [177]. Thus, we consider the work presented in the survey is a significant step towards a unified benchmark for heterogeneous tabular data for machine learning methods.

Standard datasets. There is also a need for standard evaluation datasets in the tabular data learning area. In comparison to the computer vision domain, there is no standard evaluation dataset for tabular data like MNIST [205] or ImageNet [204]. Existence of such datasets would benefit the community for efficient machine and deep learning models evaluation. A possible standard tabular dataset should contain a variety of variable types, such as different categorical features with high and low cardinality, numerical features with outliers. Additionally, since real-world tabular data often contain missing values [224], it is desirable to include variables that incorporate such elements.

Tabular data pre-processing. The heterogeneity of tabular data (e.g., categorical and sparse values) presents many challenges for deep neural networks and other machine learning algorithms. Therefore, a number of deep learning solutions transform them into a homogeneous representation more suitable for neural networks. While the additional overhead is small, such transformations can boost performance considerably and should, thus, be among the first strategies applied in real-world scenarios. Such transformations, however, may result in loss of information contained in the data [225, 226, 227]. I

Transformer architectures for deep learning on tabular data. In recent years, from an architecture perspective, Transformer-based solutions [24] have become increasingly popular [160, 158, 228, 6]. This architectures offers a number of advantages compared to standard neural network architectures, such as the ability to learn both categorical and numerical features simultaneously. In addition, self-supervised or unsupervised pre-training methods that utilize unlabeled tabular data to train parts of deep learning models are becoming increasingly popular, and are not limited to transformer-based methods. Furthermore, attention maps from Transformer models may help to interpret the outcome of models [229]. However, there is an ongoing debate regarding the viability of attention maps as explanations for the model's predictions [230, 231].

Regularization models for tabular data. As we stated in our survey (Sec. 2.1.1), regularization has also been shown to reduce the hypersensitivity of deep neural network models and to improve their overall performance [172, 83, 232]. For deep neural networks to perform more robustly and accurately on tabular data, regularization might be a crucial aspect.

Self-supervised learning for tabular data. In order to train deep neural networks, large amounts of labeled data are usually required; however, the labeling of the data is an expensive process and takes significant time [233]. Rather than employing this expensive step, self-supervised methods can be used to learn general feature representations from unlabeled data sources. These methods have also shown astonishing results in computer vision and natural language processing [40, 234]. Recent works in this direction [166, 235, 236, 160] deal with heterogeneous tabular data.

Tabular data transformation using TreeDrivenEncoder. By encoding information in a decision tree structure, the TreeDrivenEncoder procedure from our DeepTLF framework [2] creates a new homogeneous representation for heterogeneous tabular data, which can be seen as local feature selection or feature engineering. Since real-valued features are typically represented as 64-bit float types, the encoded binary data has a drastically smaller size than the original heterogeneous data, e.g., the generated Boolean values can be efficiently represented as binary vectors (i.e., 1 bit per value). As a result, the final component of the DeepTLF model can be trained efficiently. It is also possible to efficiently encode only categorical data using the TreeDrivenEncoder algorithm, and concatenate the numerical variables with the encoded vectors. Contrary to leaf-based encoding methods [182, 168], our transformation scheme produces binary features by utilizing the entire decision tree, while leaf-based encoding schemes create meta categorical features by exploiting the leaves.

Decision tree model choice for DeepTLF. Noteworthy, the proposed DeepTLF framework can be implemented with any decision tree ensemble as its basic algorithm; however, in our paper, we choose to use gradient boosting decision trees [53], which are well known for their superior performance on tabular data with heterogeneity as well as their capability of handling a wide range of features and feature value irregularities (such as missing values, outliers, etc.)[1, 154]. As part of the gradient boosting algorithm, trees are constructed sequentially, which means each tree minimizes the loss to a possible extent based on the current state. Due to the conditional dependencies between the trees in the gradient boosting ensemble, they are able to provide adequate coverage of the distribution of the training data. Furthermore, DeepTLF can be applied to unsupervised learning problems. As a possible solution, multiple variables can be used as targets in the gradient boosting decision trees algorithm after the obtained feature vectors are stacked into a single meta feature vector. As an alternative, the isolated forest algorithm [237] can be employed for the first stage of the DeepTLF model.

3.2 Synthetic Data Generation

In this Section, we provide a further discussion on the outcome of our work for heterogeneous tabular data generation using pre-trained large language models, called generation of realistic tabular data (GReaT) (Sec. 2.2).

State-of-art-performance of the GReaT method. Our proposed method GReaT outperforms state-of-the-art models for tabular data generation with respect to the machine learning efficiency and discriminator measures [102]. We improved the best existing methods by up to 44% on the California Housing dataset [176]. Regarding the discriminator measure, our generative model for tabular data outperforms the state of the art by up to 30% on the Heloc dataset [174]. We hypothesize that the superior performance of the GReaT approach comes from its ability to capture complex relationships and patterns in the data without extensive data pre-processing, allowing it to generate more realistic and diverse samples. Overall, the GReaT model represents a significant step forward in the field of tabular data generation and has the potential to improve the performance of various machine learning tasks.

Processing of numerical values with the GReaT approach. Since we convert heterogeneous tabular data into simple text-based representations such that continuous and discrete numerical values are represented as character sequences (Sec. 2.2), multiple independent studies have shown that Transformer-based models are capable of understanding and processing numerical data encoded in such a way [238, 239]. Thus, these previous observations are in agreement with the outstanding performance of GReaT. Additionally, a smarter encoding of numerical values can also be considered as a possible improvement.

Textual representation engineering. Our approach introduces new possibilities for another type of feature engineering concept [240] – which we refer to as *textual representation engineering* for tabular data. As demonstrated in the experiment (Appendix A.1), the use of feature names improves the performance of pretrained large language models. In this context, representation engineering could also be viewed as prompt creation [241]. This opens many avenues for future experiments and research.

Variable order independence. One of the key steps in the GReaT algorithm is to permute the order of the variables. In all our experiments in the presented paper, we demonstrated that a large language model is able to learn tabular data without explicitly specifying the order of the variables. This suggests that large language models are able to generalize and extract meaningful information from the data, even when it is presented in a random or unstructured manner. Furthermore, we found that for several datasets, the performance of these models improved when the data was randomly permuted. This indicates that these models are in a position to take advantage of the inherent relationships and connections within the data, in order to learn and make more accurate predictions. In general, our results demonstrate the potential of large language models to learn from tabular data and make valuable contributions in fields such as finance, healthcare, and marketing.

The importance of pre-training. Furthermore, the experiments in this dissertation, further demonstrate that pretrained large language models perform better on tabular data. We hypothesize that this is due to the fact that tabular data often contains textual information in the name or value of features, which corresponds well to the ability of large language models to capture long-term patterns and dependencies in text. It is our intention to continue exploring the use of large language models on tabular data and to investigate potential applications and improvements in this area in the future. As a result of the capability of these models to handle structured data, we believe that they have significant implications for the future of machine learning and data analysis.

Applications of the GReaT method. Based on our qualitative and qualitative evaluation of the proposed synthetic data generation algorithm, we believe that in highly critical machine learning applications, such as healthcare, where the collection of new samples is usually costly or might be impossible due to ethical concerns, the quality of the synthetically generated samples is of great importance, and thus more computational resources to generate higher quality samples could be well justified. As an illustration, synthetic data is widely used in medical ML [42]. Therefore, if realistic synthetic data is required and computation resources are not an issue, the proposed method deserves consideration.

3.3 Explainable Deep Learning

In this section, we provide a further discussion on explainability of deep learning models.

Relational local explanations. The relational local explanation (RLE) method can work with multiple data modalities, since local explanations from the RLE framework show competitive performance against selected feature attribution baselines. Overall, our quantitative experimental results for the proposed method in our manuscript resembles similar image areas or words from highlighting other state-of-the-art, non-relational explanation methods. In qualitative experiments, the proposed approach shows the best results on the selected metric.

Evaluation of the explanations. One of the key challenges in the explainable machine learning field is the evaluation of the algorithms. This problem comes from the fact that the ground truth is not available. Furthermore, some of the evaluation method for local feature attribution are computationally costly. The research within this dissertation has also contributed to the ROAD method, which is an improvement of the ROAR framework [219] that bypasses the retraining step by reducing the computation time significantly.

Interpretable deep learning models for tabular data. It is undeniable that interpretability is desirable, particularly for tabular data modeling, since it frequently contains personal data. The number of approaches that offer it out-of-the-box is increasing, but most the current deep neural network models are still primarily concerned with optimizing with respect to error-related measures. However, it is essential that deep tabular learning results be interpretable in order to fully understand model decisions and results, especially when it comes to life-critical applications. Furthermore, explanations of models can be employed as a means of identifying and mitigating potential unwanted biases and in eliminating discriminatory practices [242, 243].

The proposed CancelOut layer (Sec. 2.3.4, Appendix A.2), due to its simplicity, can be easily integrated into most existing deep learning architectures, making it a versatile and valuable addition to any deep model. By providing a global explanation of the model's predictions, the proposed layer can help improve the interpretability of deep learning models, making them more transparent and easier to understand. This can be especially useful in domains where trust and accountability are important, such as in healthcare or finance. Additionally, the global explanations provided by the CancelOut layer can be used to identify areas where the model may be performing poorly, allowing researchers and developers to improve the model and increase its accuracy. Overall, the proposed layer has the potential to greatly enhance the utility and practicality of deep learning models.

Unsupervised aggregation of feature attribution. Within the proposed aggregation method (Sec. 2.3.2) we demonstrate that an ensemble of local explanation helps to improve the performance and increase robustness of the interpretations. This opens new possibilities for the use of explainable artificial intelligence [244, 148] in complex and dynamic environments, such as healthcare or finance. By combining the insights from multiple local explanations, the aggregation method allows for a more comprehensive and nuanced understanding of the model's decision-making process. Furthermore, the aggregation method can be applied to various types of explanations, such as saliency maps for visual data and perturbation-based explanations for tabular data. This versatility allows for flexibility in the choice of explanation techniques, depending on the specific task at hand and the needs of the user. In addition, the aggregation method can be easily integrated into existing explainable machine learning frameworks and tools, making it a practical and accessible solution to improve the interpretability of machine and deep models. This has the potential to enhance the trust and transparency of data-driven systems, which are crucial for their successful deployment in real-world applications.

3.4 Outlook and Future Work

Although we have addressed a number of issues in the deep tabular learning and explainability domains, there are still a variety of open questions that need to be addressed in the future. For example, how can we effectively incorporate domain knowledge into deep tabular models to improve their performance and interpretability? Can we ensure that the explanations provided by these models are accurate and trustworthy?

For the deep tabular learning domain, in particular, a novel deep learning architecture that takes into account the tabular data heterogeneity and structure needs to be devised. As a result, data-driven, low-latency systems that work with tabular data could benefit from this approach. Due to the difficulty of scaling the current decision tree-based approaches, there is a need to develop new methods.

Self-supervised learning on tabular data. Another point of interest for the future research could be the adaption of self-supervised learning schemes for tabular data, where deep models can learn useful data representations. We believe that the proposed DeepTLF framework can be used in this setting; however, for the data encoding step, unsupervised decision tree algorithms may be of use.

Novel evaluation measures for synthetic tabular data. Considering the generation of tabular data, there is a need for reliable and consistent measures that take into account semantic coherence when generating tabular data. The process can be accomplished by learning combinations of categorical variables and numerical ranges for corresponding features. By utilizing such a measure, synthetic data can be provided with greater assurance.

Multi-modal data generation. Along with the usual numerical values, tabular data frequently contains textual metadata, e.g., feature names, named categories (“male”, “female”), and open text features (e.g., `remarks`). The recent evolution of transformer neural networks now permits to holistically unite this information, which was processed separately in the past, and learn context-specific, robust, and meaningful representations. In the case of tabular data, information comes from the textual and numerical data modalities. However, progress is not halted at this point: The recent advances in vision transformers [245]) suggest possible extensions to the domain of image data, which could possibly also be encoded in tokens and be processed by a transformer network as part of a multi-modal token sequence. Hence, we argue that by providing additional textual (and semantically meaningful) information about the data, modern large-scale language models should be able to model the data more realistically and robustly.

Efficient Transformer models for tabular data. A clear limitation associated with Transformer-based methods is the computational complexity, due to a high number of learning parameters, e.g., weights [246]. Since the GReaT approach is based on such architectures, it requires more time for the fine-tuning step than other methods based on GAN or VAE models. Therefore, in future work, one can look for more efficient generative language models dedicated to the heterogeneous tabular data.

Transfer learning for tabular data. Reusing knowledge gained from solving one problem and adapting it to a different objective is the research problem addressed by transfer learning. Although transfer learning is successful in computer vision and natural language processing applications [41, 247], there are no efficient and generally accepted methods for transfer learning with tabular data. In this regard, there have been recent developments [248]. One of the possible ways is to utilize the Transformer-based neural network architecture since they usually do not have the input share restriction as convolutional or recurrent neural networks.

In the case of explainability of machine and deep learning models, several major topics are worth investigating.

Reliable evaluation of feature attribution algorithms. In addition to reliable and robust explanations, there must also be trustworthy evaluations of feature attributions. Another point of the discussed manuscripts’ continuation with (relational) local explanations is the absence of evaluation metrics. In the absence of access to ground truth, establishing a reliable and plausible measure will prove to be challenging. On the other hand, with an unambiguous measure, a possible strategy would involve direct optimization of it.

Robust local explanations via realistic auxiliary data. One of the main principles of permutation-based feature attribution methods, such as LIME [118] and SHAP [119], is to approximate a single data point by creating artificial points around it. However, this approach is vulnerable to adversarial attacks, as shown in [249]. One potential defense against these attacks is to use more realistic auxiliary data samples. In future work, it may be possible to use the GReaT method 2.2 to generate more realistic samples around the target data point, providing a more robust explanation.

Relational local explanations for tabular data. Furthermore, further research into extending our framework for explaining relationships in heterogeneous tabular data would be beneficial. One potential approach could be to construct a graph using both categorical and binned numerical variables. This would enable a more comprehensive analysis of the relationships between different data points. Additionally, utilizing the relational local explanation technique could provide insights into the complex connections between various variables in heterogeneous tabular data. Thus, this extension of our framework has the potential to greatly enhance our understanding of heterogeneous data sets.

Probabilistic explanations. Most current approaches for local or global explanations are deterministic, offering only a point estimate. As a result, assessing the specificity of these methods can be challenging. One potential solution involves integrating a probabilistic perspective into local explanations. By examining distribution rather than single points, we can better quantify the uncertainties inherent in the explanation model. This enables a more comprehensive understanding of the underlying mechanisms and factors leading to a particular prediction or decision, along with a clearer assessment of its uncertainty and potential biases. This approach can be especially valuable in high-stakes situations where the precision and reliability of explanations are crucial for ensuring decision safety and effectiveness [250, 251]. In future research, both local and global explanations could benefit from the integration of probabilistic models, resulting in a more robust and accurate framework for understanding prediction and decision-driving factors. Moreover, this probabilistic approach could be easily incorporated into decision-making processes, facilitating the quantification and management of uncertainty in high-stakes situations. This development could represent a significant step towards creating more transparent and reliable machine learning models, thereby better supporting decision-making across a range of applications. Future work should concentrate on the creation of probabilistic models for local and global explanations that can be seamlessly integrated into existing machine learning frameworks, as well as on the evaluation of these models' effectiveness in real-world scenarios.

To conclude this Chapter, I would like to use the following quote from Alan M. Turing [252]: *"We can only see a short distance ahead, but we can see plenty there that needs to be done."*

Chapter 4

Conclusion

As a result of the research presented in this thesis, a valuable contribution has been made to scientific knowledge that is related to heterogeneous tabular data modeling and generation, as well as the explanation of machine learning algorithms.

In the context of deep neural networks and tabular data, we have summarized and evaluated existing state-of-the-art deep learning approaches for tabular data. We have also proposed a new hybrid model that combines the preprocessing capabilities of decision tree ensemble algorithms with the flexibility of neural networks. Furthermore, we have introduced a novel approach for synthetic tabular data generation using large language models, which outperforms the current state-of-the-art by a significant margin. This approach will make research on deep learning and tabular data more accessible for various critical domains, such as healthcare, finance, and other application areas related to everyday life.

Furthermore, one of the main challenges when it comes to the application of deep neural networks is their lack of explainability. It is, therefore, vital that explainable machine learning techniques be applied in order to improve the interpretability of the results as well as to ensure that the model is trustworthy. To this end, in the present work, we have addressed the current issues of explainable machine learning methods and have devised approaches for robust and stable local explanations, as well as, global explanations for deep neural networks.

Finally, the findings presented in this dissertation can serve as a basis for future research in the area of heterogeneous tabular data modeling, as well as for robust and explainable machine learning methods. Overall, the outcomes of our studies open up new possibilities for future research on tabular data and the explainability of deep neural networks to achieve previously unattainable results.

Bibliography

[1] Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. Deep neural networks and tabular data: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2022. (Cited on pages xix, 2, 3, 6, 7, 8, 9, 10, 19, 20, 24, 27, and 43.)

[2] Vadim Borisov, Klaus Broelemann, Enkelejda Kasneci, and Gjergji Kasneci. Deeptlf: robust deep neural networks for heterogeneous tabular data. *International Journal of Data Science and Analytics*, pages 1–16, 2022. (Cited on pages xix and 43.)

[3] Vadim Borisov, Johannes Haug, and Gjergji Kasneci. CancelOut: A layer for feature selection in deep neural networks. In *International Conference on Artificial Neural Networks*, pages 72–83. Springer, 2019. (Cited on pages xix and 15.)

[4] Vadim Borisov, Johannes Meier, Johan van den Heuvel, Hamed Jalali, and Gjergji Kasneci. A robust unsupervised ensemble of feature-based explanations using restricted boltzmann machines. *arXiv preprint arXiv:2111.07379*, 2021. (Cited on pages xix, 32, and 34.)

[5] Yao Rong, Tobias Leemann, Vadim Borisov, Gjergji Kasneci, and Enkelejda Kasneci. A consistent and efficient evaluation strategy for attribution methods. In *International Conference on Machine Learning*, pages 18770–18795. PMLR, 2022. (Cited on pages xx and 35.)

[6] Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. Language models are realistic tabular data generators. *arXiv preprint arXiv:2210.06280*, 2022. (Cited on pages xx, 27, and 42.)

[7] Vadim Borisov and Gjergji Kasneci. Relational local explanations. *arXiv preprint arXiv:2212.12374*, 2022. (Cited on page xx.)

[8] Vadim Borisov, Enkelejda Kasneci, and Gjergji Kasneci. Robust cognitive load detection from wrist-band sensors. *Computers in Human Behavior Reports*, 4:100116, 2021. (Cited on page xx.)

[9] Michael Gröger, Vadim Borisov, and Gjergji Kasneci. Boxshrink: From bounding boxes to segmentation masks. In *Workshop on Medical Image Learning with Limited and Noisy Data*, pages 65–75. Springer, 2022. (Cited on page xx.)

[10] Nikolaos Nikolaou, Ingo P Waldmann, Angelos Tsiaras, Mario Morvan, Billy Edwards, Kai Hou Yip, Giovanna Tinetti, Subhajit Sarkar, James M Dawson, Vadim Borisov, et al. Lessons learned from the 1st ariel machine learning challenge: Correcting transiting exoplanet light curves for stellar spots. *arXiv preprint arXiv:2010.15996*, 2020. (Cited on page xxi.)

[11] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015. (Cited on pages 1 and 24.)

[12] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. (Cited on pages 1, 4, and 24.)

[13] Emmanuel Gbenga Dada, Joseph Stephen Bassi, Haruna Chiroma, Adebayo Olusola Adetunmbi, Opeyemi Emmanuel Ajibuwa, et al. Machine learning for email spam filtering: review, approaches and open research problems. *Heliyon*, 5(6):e01802, 2019. (Cited on page 1.)

[14] Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Phrase-based & neural unsupervised machine translation. *arXiv preprint arXiv:1804.07755*, 2018. (Cited on page 1.)

[15] Mark F Hansen, Melvyn L Smith, Lyndon N Smith, Michael G Salter, Emma M Baxter, Marianne Farish, and Bruce Grieve. Towards on-farm pig face recognition using convolutional neural networks. *Computers in Industry*, 98:145–152, 2018. (Cited on page 1.)

[16] Byung-Hak Kim, Ethan Vizitei, and Varun Ganapathi. Gritnet: Student performance prediction with deep learning. *arXiv preprint arXiv:1804.07405*, 2018. (Cited on page 1.)

[17] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *International journal of computer vision*, 128(2):261–318, 2020. (Cited on pages 1 and 12.)

[18] Artúr István Károly, Péter Galambos, József Kuti, and Imre J Rudas. Deep learning in robotics: Survey on model structures and training strategies. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51(1):266–279, 2020. (Cited on page 1.)

[19] Sorin Grigorescu, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu. A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics*, 37(3):362–386, 2020. (Cited on page 1.)

[20] John D Kelleher. *Deep learning*. MIT press, 2019. (Cited on page 1.)

[21] Andreas Zell. *Simulation neuronaler netze*, volume 1. Addison-Wesley Bonn, 1994. (Cited on page 1.)

[22] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. (Cited on page 1.)

[23] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. (Cited on page 1.)

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. (Cited on pages 1, 4, 9, 12, and 42.)

[25] Ravid Shwartz-Ziv and Amitai Armon. Tabular Data: Deep Learning is Not All You Need. *arXiv preprint arXiv:2106.03253*, 2021. (Cited on pages 1, 7, and 41.)

[26] Dennis Ulmer, Lotta Meijerink, and Giovanni Cinà. Trust issues: Uncertainty estimation does not enable reliable ood detection on medical tabular data. In *Machine Learning for Health*, pages 341–354. PMLR, 2020. (Cited on page 1.)

[27] Jillian M Clements, Di Xu, Nooshin Yousefi, and Dmitry Efimov. Sequential deep learning for credit risk monitoring with tabular financial data. *arXiv preprint arXiv:2012.15330*, 2020. (Cited on page 1.)

[28] Ying Zhang, Mutahar Safdar, Jiarui Xie, Jinghao Li, Manuel Sage, and Yaoyao Fiona Zhao. A systematic review on data of additive manufacturing for machine learning applications: the data quality, type, preprocessing, and management. *Journal of Intelligent Manufacturing*, pages 1–36, 2022. (Cited on page 1.)

[29] Jiantao Wu, Fabrizio Orlandi, Declan O'Sullivan, Enrico Pisoni, and Soumyabrata Dev. Boosting climate analysis with semantically uplifted knowledge graphs. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 15:4708–4718, 2022. (Cited on page 1.)

[30] Tom Shenkar and Lior Wolf. Anomaly detection for tabular data with internal contrastive learning. In *International Conference on Learning Representations*, 2021. (Cited on page 1.)

[31] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR)*, 52(1):1–38, 2019. (Cited on page 1.)

[32] Anna L Buczak and Erhan Guven. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications surveys & tutorials*, 18(2):1153–1176, 2015. (Cited on page 1.)

[33] Adir Even, Ganesan Shankaranarayanan, and Paul D Berger. Economics-driven data management: An application to the design of tabular data sets. *IEEE Transactions on Knowledge and Data Engineering*, 19(6):818–831, 2007. (Cited on page 1.)

[34] Changgwon Dang, Taejeong Choi, Seungsoo Lee, Soohyun Lee, Mahboob Alam, Mina Park, Seungkyu Han, Jaegu Lee, and Duytang Hoang. Machine learning-based live weight estimation for hanwoo cow. *Sustainability*, 14(19):12661, 2022. (Cited on page 1.)

[35] Andrea Brunello, Angelo Montanari, and Nicola Saccomanno. A genetic programming approach to wifi fingerprint meta-distance learning. *Pervasive and Mobile Computing*, 85:101681, 2022. (Cited on page 1.)

[36] Thomas Veran, Pierre-Edouard Portier, and François Fouquet. Interpretable hierarchical symbolic regression for safety-critical systems with an application to highway crash prediction. *Engineering Applications of Artificial Intelligence*, 117:105534, 2023. (Cited on page 1.)

[37] Langcheng Zhao, Fenglin Zhang, Huanhuan Zhang, Yumeng Liang, Anfu Zhou, and Huadong Ma. Robust respiratory rate monitoring using smartwatch photoplethysmography. *IEEE Internet of Things Journal*, 2022. (Cited on page 1.)

[38] Christian Janiesch, Patrick Zschech, and Kai Heinrich. Machine learning and deep learning. *Electronic Markets*, 31(3):685–695, 2021. (Cited on page 1.)

[39] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal machine learning: A survey and taxonomy. *IEEE transactions on pattern analysis and machine intelligence*, 41(2):423–443, 2018. (Cited on pages 1, 8, and 25.)

[40] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):4037–4058, 2020. (Cited on pages 1 and 43.)

[41] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In *International conference on artificial neural networks*, pages 270–279. Springer, 2018. (Cited on pages 1 and 47.)

[42] Mikel Hernandez, Gorka Epelde, Ane Alberdi, Rodrigo Cilla, and Debbie Rankin. Synthetic data generation for tabular health records: A systematic review. *Neurocomputing*, 2022. (Cited on pages 2, 8, and 45.)

[43] Edward Choi, Siddharth Biswal, Bradley Malin, Jon Duke, Walter F. Stewart, and Jimeng Sun. Generating Multi-label Discrete Patient Records using Generative

Adversarial Networks. *Machine learning for healthcare conference*, pages 286–305, 2017. (Cited on pages 2 and 9.)

[44] Feiyu Xu, Hans Uszkoreit, Yangzhou Du, Wei Fan, Dongyan Zhao, and Jun Zhu. Explainable ai: A brief survey on history, research areas, approaches and challenges. In *CCF international conference on natural language processing and Chinese computing*, pages 563–574. Springer, 2019. (Cited on pages 2, 12, and 35.)

[45] Agneza Krajna, Mihael Kovac, Mario Brcic, and Ana Šarčević. Explainable artificial intelligence: An updated perspective. In *2022 45th Jubilee International Convention on Information, Communication and Electronic Technology (MIPRO)*, pages 859–864. IEEE, 2022. (Cited on pages 2, 12, and 35.)

[46] Amina Adadi and Mohammed Berrada. Peeking inside the black-box: a survey on explainable artificial intelligence (xai). *IEEE access*, 6:52138–52160, 2018. (Cited on pages 2, 12, and 35.)

[47] Arun Das and Paul Rad. Opportunities and challenges in explainable artificial intelligence (xai): A survey. *arXiv preprint arXiv:2006.11371*, 2020. (Cited on pages 2, 12, 15, and 35.)

[48] Maria Sahakyan, Zeyar Aung, and Talal Rahwan. Explainable artificial intelligence for tabular data: A survey. *IEEE Access*, 2021. (Cited on pages 2, 4, and 12.)

[49] Ning Xie, Gabrielle Ras, Marcel van Gerven, and Derek Doran. Explainable deep learning: A field guide for the uninitiated. *arXiv preprint arXiv:2004.14545*, 2020. (Cited on pages 2, 12, 13, 34, and 35.)

[50] Ravid Shwartz-Ziv and Amitai Armon. Tabular data: Deep learning is not all you need. *Information Fusion*, 81:84–90, 2022. (Cited on page 3.)

[51] Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. *arXiv preprint arXiv:2106.11959*, 2021. (Cited on pages 3 and 7.)

[52] Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. Why do tree-based models still outperform deep learning on tabular data? *arXiv preprint arXiv:2207.08815*, 2022. (Cited on pages 3 and 7.)

[53] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001. (Cited on pages 3, 19, and 43.)

[54] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, 2020. (Cited on page 4.)

[55] Deepak Soekhoe, Peter van der Putten, and Aske Plaat. On the impact of data set size in transfer learning using deep neural networks. In *International symposium on intelligent data analysis*, pages 50–60. Springer, 2016. (Cited on page 4.)

[56] Yuji Roh, Geon Heo, and Steven Euijong Whang. A survey on data collection for machine learning: a big data-ai integration perspective. *IEEE Transactions on Knowledge and Data Engineering*, 33(4):1328–1347, 2019. (Cited on page 4.)

[57] Yifan Li, Xiaohui Yu, and Nick Koudas. Data acquisition for improving machine learning models. *arXiv preprint arXiv:2105.14107*, 2021. (Cited on page 4.)

[58] Emily Namey, Greg Guest, Amy O'Regan, Christine L Godwin, Jamilah Taylor, and Andres Martinez. How does mode of qualitative data collection affect data and cost? findings from a quasi-experimental study. *Field Methods*, 32(1):58–74, 2020. (Cited on page 4.)

[59] W Nicholson Price and I Glenn Cohen. Privacy in the age of medical big data. *Nature medicine*, 25(1):37–43, 2019. (Cited on page 4.)

[60] Adrien Bibal, Michael Lognoul, Alexandre De Streel, and Benoît Frénay. Legal requirements on explainability in machine learning. *Artificial Intelligence and Law*, 29(2):149–169, 2021. (Cited on page 4.)

[61] Anastasiya Kiseleva, Dimitris Kotzinos, and Paul De Hert. Transparency of ai in healthcare as a multilayered system of accountabilities: Between legal requirements and technical limitations. *Frontiers in artificial intelligence*, 5, 2022. (Cited on page 4.)

[62] CA OAG. Ccpa regulations: Final regulation text. *Office of the Attorney General, California Department of Justice*, 2021. (Cited on pages 4 and 12.)

[63] GDPR. Regulation (eu) 2016/679 of the european parliament and of the council. *Official Journal of the European Union*, 2016. (Cited on pages 4 and 12.)

[64] Bruno Iochins Grisci, Mathias J. Krause, and Marcio Dorn. Relevance aggregation for neural networks interpretability and knowledge discovery on tabular data. *Information Sciences*, 559:111–129, 2021. (Cited on page 4.)

[65] Umang Bhatt, Alice Xiang, Shubham Sharma, Adrian Weller, Ankur Taly, Yunhan Jia, Joydeep Ghosh, Ruchir Puri, José MF Moura, and Peter Eckersley. Explainable machine learning in deployment. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pages 648–657, 2020. (Cited on page 4.)

[66] Rishi Bommasani, Drew Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney Arx, Michael Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Davis, Dora Demszky, and Percy Liang. On the opportunities and risks of foundation models. 08 2021. (Cited on page 4.)

[67] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *International conference on machine learning*, pages 7354–7363. PMLR, 2019. (Cited on page 4.)

[68] John T Hancock and Taghi M Khoshgoftaar. Survey on categorical data for neural networks. *Journal of Big Data*, 7:1–41, 2020. (Cited on page 5.)

[69] Paul Peseux, Maxime Berar, Thierry Paquet, and Victor Nicollet. Stochastic gradient descent with gradient estimator for categorical features. *arXiv preprint arXiv:2209.03771*, 2022. (Cited on page 5.)

[70] Rice University David M. Lane. *Introduction to Statistics*. David Lane, 2003. (Cited on page 5.)

[71] Mark Ryan. *Deep learning with structured data*. Simon and Schuster, 2020. (Cited on page 5.)

[72] Anantaa Kotal, Aritran Piplai, Sai Sree Laya Chukkapalli, and Anupam Joshi. Privetab: Secure and privacy-preserving sharing of tabular data. In *Proceedings of the 2022 ACM on International Workshop on Security and Privacy Analytics*, pages 35–45, 2022. (Cited on page 6.)

[73] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, 267:1–38, 2019. (Cited on page 6.)

[74] Been Kim, Rajiv Khanna, and Oluwasanmi O Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. *Advances in neural information processing systems*, 29, 2016. (Cited on page 6.)

[75] Ribana Roscher, Bastian Bohn, Marco F Duarte, and Jochen Garcke. Explainable machine learning for scientific insights and discoveries. *Ieee Access*, 8:42200–42216, 2020. (Cited on page 7.)

[76] Diogo V Carvalho, Eduardo M Pereira, and Jaime S Cardoso. Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8(8):832, 2019. (Cited on pages 7 and 16.)

[77] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. (Cited on pages 7, 21, and 26.)

[78] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001. (Cited on pages 7, 19, and 21.)

[79] Jerome H Friedman. Stochastic gradient boosting. *Computational statistics & data analysis*, 38(4):367–378, 2002. (Cited on page 7.)

[80] Abijah J Miles. The sunstroke epidemic of cincinnati, ohio, during the summer of 1881. *Public health papers and reports*, 7:293, 1881. (Cited on page 7.)

[81] Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936. (Cited on page 7.)

[82] Omar Benjelloun, Shiyu Chen, and Natasha Noy. Google dataset search by the numbers. In *International Semantic Web Conference*, pages 667–682. Springer, 2020. (Cited on page 7.)

[83] Arlind Kadra, Marius Lindauer, Frank Hutter, and Josif Grabocka. Well-tuned Simple Nets Excel on Tabular Datasets. In *Advances in Neural Information Processing Systems*, 2021. (Cited on pages 7, 41, and 42.)

[84] Liran Katzir, Gal Elidan, and Ran El-Yaniv. Net-DNF: Effective deep modeling of tabular data. In *International Conference on Learning Representations*, 2020. (Cited on page 7.)

[85] Denis Baylor, Eric Breck, Heng-Tze Cheng, Noah Fiedel, Chuan Yu Foo, Zakaria Haque, Salem Haykal, Mustafa Ispir, Vihan Jain, Levent Koc, et al. TFX: A tensorflow-based production-scale machine learning platform. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1387–1395, 2017. (Cited on page 9.)

[86] Samuel A Assefa, Danial Dervovic, Mahmoud Mahfouz, Robert E Tillman, Prashant Reddy, and Manuela Veloso. Generating synthetic data in finance: opportunities, challenges and pitfalls. In *Proceedings of the First ACM International Conference on AI in Finance*, pages 1–8, 2020. (Cited on pages 9 and 10.)

[87] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *ICLR*, 2013. (Cited on pages 9 and 11.)

[88] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014. (Cited on pages 9 and 11.)

[89] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. (Cited on page 9.)

[90] Derek Snow. Deltapy: A framework for tabular data augmentation in python. *Available at SSRN 3582219*, 2020. (Cited on page 9.)

[91] Stavroula Bourou, Andreas El Saer, Terpsichori-Helen Velivassaki, Artemis Voulkidis, and Theodore Zahariadis. A review of tabular data synthesis using gans on an ids dataset. *Information*, 12(09):375, 2021. (Cited on page 9.)

[92] Haipeng Chen, Sushil Jajodia, Jing Liu, Noseong Park, Vadim Sokolov, and VS Subrahmanian. Faketables: Using gans to generate functional dependency preserving tables with bounded real data. In *IJCAI*, pages 2074–2080, 2019. (Cited on page 9.)

[93] Lovedeep Gondara and Ke Wang. Mida: Multiple imputation using denoising autoencoders. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 260–272. Springer, 2018. (Cited on page 9.)

[94] Ramiro Daniel Camino, Christian Hammerschmidt, et al. Working with deep generative models and tabular data imputation. *ICML 2020 Artemiss Workshop*, 2020. (Cited on page 9.)

[95] Justin Engelmann and Stefan Lessmann. Conditional wasserstein gan-based oversampling of tabular data for imbalanced learning. *Expert Systems with Applications*, 174:114582, 2021. (Cited on page 9.)

[96] Matias Quintana and Clayton Miller. Towards class-balancing human comfort datasets with gans. In *Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pages 391–392, 2019. (Cited on page 9.)

[97] Aki Koivu, Mikko Sairanen, Antti Airola, and Tapio Pahikkala. Synthetic minority oversampling of vital statistics data with generative adversarial networks. *Journal of the American Medical Informatics Association*, 27(11):1667–1674, 2020. (Cited on page 9.)

[98] Sajad Darabi and Yotam Elor. Synthesising multi-modal minority samples for tabular data. *arXiv preprint arXiv:2105.08204*, 2021. (Cited on page 9.)

[99] Ju Fan, Junyou Chen, Tongyu Liu, Yuwei Shen, Guoliang Li, and Xiaoyong Du. Relational data synthesis using generative adversarial networks: A design space exploration. *Proc. VLDB Endow.*, 13(12):1962–1975, July 2020. (Cited on page 9.)

[100] Sanket Kamthe, Samuel Assefa, and Marc Deisenroth. Copula flows for synthetic data generation. *arXiv preprint arXiv:2101.00598*, 2021. (Cited on page 9.)

[101] Akash Srivastava, Lazar Valkov, Chris Russell, Michael U Gutmann, and Charles Sutton. Veegan: Reducing mode collapse in gans using implicit variational learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 3310–3320, 2017. (Cited on page 10.)

[102] Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional GAN. In *Advances in Neural Information Processing Systems*, volume 33, 2019. (Cited on pages 10, 11, 27, 28, 29, and 43.)

[103] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017. (Cited on page 10.)

[104] Adrián Sánchez-Morales, José-Luis Sancho-Gómez, Juan-Antonio Martínez-García, and Aníbal R Figueiras-Vidal. Improving deep learning performance with missing values via deletion and compensation. *Neural Computing and Applications*, 32(17):13233–13244, 2020. (Cited on page 10.)

[105] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. Deep learning for anomaly detection: A review. *ACM Computing Surveys (CSUR)*, 54(2):1–38, 2021. (Cited on page 10.)

[106] Alan F Karr, Ashish P Sanil, and David L Banks. Data quality: A statistical perspective. *Statistical Methodology*, 3(2):137–173, 2006. (Cited on page 10.)

[107] Lei Xu and Kalyan Veeramachaneni. Synthesizing Tabular Data using Generative Adversarial Networks. *arXiv preprint arXiv:1811.11264*, 2018. (Cited on page 10.)

[108] Ramesh A Dandekar and Lawrence H Cox. Synthetic tabular data: an alternative to complementary cell suppression. *Manuscript, Energy Information Administration, US*, page 4, 2002. (Cited on page 10.)

[109] Max Baak, Simon Brugman, Ilan Fridman Rojas, Lorraine Dalmeida, Ralph EQ Urlus, and Jean-Baptiste Oger. Synthsonic: Fast, probabilistic modeling and synthesis of tabular data. In *International Conference on Artificial Intelligence and Statistics*, pages 4747–4763. PMLR, 2022. (Cited on page 10.)

[110] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002. (Cited on page 11.)

[111] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, et al. Recent advances in convolutional neural networks. *Pattern Recognition*, 77:354–377, 2018. (Cited on page 12.)

[112] Shervin Minaee, Yuri Y Boykov, Fatih Porikli, Antonio J Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2021. (Cited on page 12.)

[113] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. (Cited on page 12.)

[114] Wafaa S El-Kassas, Cherif R Salama, Ahmed A Rafea, and Hoda K Mohamed. Automatic text summarization: A comprehensive survey. *Expert Systems with Applications*, 165:113679, 2021. (Cited on page 12.)

[115] Marouane Birjali, Mohammed Kasri, and Abderrahim Beni-Hssane. A comprehensive survey on sentiment analysis: Approaches, challenges and trends. *Knowledge-Based Systems*, 226:107134, 2021. (Cited on page 12.)

[116] Tjeerd AJ Schoonderwoerd, Wiard Jorritsma, Mark A Neerincx, and Karel Van Den Bosch. Human-centered xai: Developing design patterns for explanations of clinical decision support systems. *International Journal of Human-Computer Studies*, 154:102684, 2021. (Cited on page 12.)

[117] Hanxiong Chen, Xu Chen, Shaoyun Shi, and Yongfeng Zhang. Generate natural language explanations for recommendation. *arXiv preprint arXiv:2101.03392*, 2021. (Cited on page 13.)

[118] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should I trust you? explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016. (Cited on pages 13, 14, 33, and 47.)

[119] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc., 2017. (Cited on pages 13, 14, 33, and 47.)

[120] Sahra Ghalebikesabi, Lucile Ter-Minassian, Karla Diaz-Ordaz, and Chris Holmes. On locality of local explanation models. *arXiv preprint arXiv:2106.14648*, 2021. (Cited on page 13.)

[121] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017. (Cited on page 13.)

[122] Oscar Li, Hao Liu, Chaofan Chen, and Cynthia Rudin. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. (Cited on page 13.)

[123] Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim. Towards automatic concept-based explanations. *Advances in Neural Information Processing Systems*, 32, 2019. (Cited on page 13.)

[124] Wojciech Samek, Grégoire Montavon, Sebastian Lapuschkin, Christopher J Anders, and Klaus-Robert Müller. Explaining deep neural networks and beyond: A review of methods and applications. *Proceedings of the IEEE*, 109(3):247–278, 2021. (Cited on page 13.)

[125] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. (Cited on page 13.)

[126] Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. *arXiv preprint arXiv:1711.06104*, 2017. (Cited on page 14.)

[127] Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2018. (Cited on page 14.)

[128] Gjergji Kasneci and Thomas Gottron. Licon: A linear weighting scheme for the contribution ofinput variables in deep artificial neural networks. In *CIKM*, 2016. (Cited on page 14.)

[129] Yi Wei, Ming-Ching Chang, Yiming Ying, Ser Nam Lim, and Siwei Lyu. Explain black-box image classifications using superpixel-based interpretation. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 1640–1645. IEEE, 2018. (Cited on pages 14 and 15.)

[130] Beomsu Kim, Junghoon Seo, Seunghyeon Jeon, Jamyoung Koo, Jeongyeol Choe, and Taegyun Jeon. Why are saliency maps noisy? cause of and solution to noisy saliency maps. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 4149–4157. IEEE, 2019. (Cited on page 14.)

[131] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *ICML*, 2018. (Cited on page 14.)

[132] Pascal Sturmfels, Scott Lundberg, and Su-In Lee. Visualizing the impact of feature attribution baselines. *Distill*, 5(1):e22, 2020. (Cited on page 14.)

[133] Harshay Shah, Prateek Jain, and Praneeth Netrapalli. Do input gradients highlight discriminative features? *arXiv preprint arXiv:2102.12781*, 2021. (Cited on page 14.)

[134] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. *arXiv preprint arXiv:1810.03292*, 2018. (Cited on page 14.)

[135] Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. Layer-wise relevance propagation: an overview. *Explainable AI: interpreting, explaining and visualizing deep learning*, pages 193–209, 2019. (Cited on page 14.)

[136] Jindong Gu, Yinchong Yang, and Volker Tresp. Understanding individual decisions of CNNs via contrastive backpropagation. In *Asian Conference on Computer Vision*, pages 119–134. Springer, 2018. (Cited on page 14.)

[137] Brian Kenji Iwana, Ryohei Kuroki, and Seiichi Uchida. Explaining convolutional neural networks using softmax gradient layer-wise relevance propagation. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 4176–4185. IEEE, 2019. (Cited on page 14.)

[138] Leon Sixt, Maximilian Granz, and Tim Landgraf. When explanations lie: Why many modified bp attributions fail. In *International Conference on Machine Learning*, pages 9046–9057. PMLR, 2020. (Cited on page 14.)

[139] Marko Robnik-Šikonja and Marko Bohanec. Perturbation-based explanations of prediction models. In *Human and machine learning*, pages 159–175. Springer, 2018. (Cited on page 14.)

[140] Damien Garreau and Ulrike Luxburg. Explaining the explainer: A first theoretical analysis of lime. In *International Conference on Artificial Intelligence and Statistics*, pages 1287–1296. PMLR, 2020. (Cited on page 14.)

[141] Damien Garreau and Dina Mardaoui. What does lime really see in images? *arXiv preprint arXiv:2102.06307*, 2021. (Cited on page 14.)

[142] Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. *arXiv preprint arXiv:1806.07421*, 2018. (Cited on page 14.)

[143] Ruth Fong, Mandela Patrick, and Andrea Vedaldi. Understanding deep networks via extremal perturbations and smooth masks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2950–2958, 2019. (Cited on page 14.)

[144] Ludwig Schallner, Johannes Rabold, Oliver Scholz, and Ute Schmid. Effect of superpixel aggregation on explanations in lime–a case study with biological data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 147–158. Springer, 2019. (Cited on page 15.)

[145] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P Trevino, Jiliang Tang, and Huan Liu. Feature selection: A data perspective. *ACM computing surveys (CSUR)*, 50(6):1–45, 2017. (Cited on page 15.)

[146] Madhumita Sushil, Simon Šuster, and Walter Daelemans. Rule induction for global explanation of trained models. *arXiv preprint arXiv:1808.09744*, 2018. (Cited on page 15.)

[147] Krzysztof Grabczewski and Norbert Jankowski. Feature selection with decision tree criterion. In *Fifth International Conference on Hybrid Intelligent Systems (HIS'05)*, pages 6–pp. IEEE, 2005. (Cited on page 15.)

[148] David Gunning, Mark Stefik, Jaesik Choi, Timothy Miller, Simone Stumpf, and Guang-Zhong Yang. Xai—explainable artificial intelligence. *Science robotics*, 4(37):eaay7120, 2019. (Cited on pages 16 and 46.)

[149] An Phi Nguyen and María Rodríguez Martínez. On quantitative aspects of model interpretability. *arXiv preprint arXiv:2007.07584*, 2020. (Cited on pages 16 and 35.)

[150] Peter Hase and Mohit Bansal. Evaluating explainable AI: Which algorithmic explanations help users predict model behavior? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5540–5552, 2020. (Cited on pages 16 and 35.)

[151] Chih-Kuan Yeh, Cheng-Yu Hsieh, Arun Suggala, David I Inouye, and Pradeep K Ravikumar. On the (in) fidelity and sensitivity of explanations. *Advances in Neural Information Processing Systems*, 32:10967–10978, 2019. (Cited on pages 16 and 35.)

[152] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017. (Cited on page 16.)

[153] Yao Rong, Tobias Leemann, Thai-trang Nguyen, Lisa Fiedler, Tina Seidel, Gjergji Kasneci, and Enkelejda Kasneci. Towards human-centered explainable ai: User

studies for model explanations. *arXiv preprint arXiv:2210.11584*, 2022. (Cited on page 16.)

[154] Casper Solheim Bojer and Jens Peder Meldgaard. Kaggle forecasting competitions: An overlooked learning opportunity. *International Journal of Forecasting*, 37(2):587–603, 2021. (Cited on pages 19 and 43.)

[155] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 785–794, 2016. (Cited on pages 19, 21, and 23.)

[156] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in neural information processing systems*, pages 3146–3154, 2017. (Cited on pages 19, 21, and 23.)

[157] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. In *Advances in neural information processing systems*, pages 6638–6648, 2018. (Cited on pages 19, 21, and 23.)

[158] Sercan O Arik and Tomas Pfister. TabNet: Attentive interpretable tabular learning. *arxiv:1908.07442*, 2019. (Cited on pages 20, 21, 22, and 42.)

[159] Xin Huang, Ashish Khetan, Milan Cvitkovic, and Zohar Karnin. TabTransformer: Tabular Data Modeling Using Contextual Embeddings. *arxiv:2012.06678*, 2020. (Cited on pages 20 and 21.)

[160] Gowthami Somepalli, Micah Goldblum, Avi Schwarzschild, C Bayan Bruss, and Tom Goldstein. SAINT: Improved neural networks for tabular data via row attention and contrastive pre-training. *arXiv preprint arXiv:2106.01342*, 2021. (Cited on pages 21, 23, 42, and 43.)

[161] Scott Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *NeurIPS*, 2017. (Cited on page 21.)

[162] Evelyn Fix. *Discriminatory analysis: nonparametric discrimination, consistency properties*. USAF school of Aviation Medicine, 1951. (Cited on page 21.)

[163] Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. *Classification and regression trees*. Routledge, 2017. (Cited on page 21.)

[164] Klaus Broelemann and Gjergji Kasneci. A gradient-based split criterion for highly accurate and transparent model trees. In *IJCAI*, 2019. (Cited on page 21.)

[165] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943. (Cited on page 21.)

[166] Jinsung Yoon, Yao Zhang, James Jordon, and Mihaela van der Schaar. Vime: Extending the success of self-and semi-supervised learning to tabular domaindim. *Advances in Neural Information Processing Systems*, 33, 2020. (Cited on pages 21 and 43.)

[167] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. Deepfm: a factorization-machine based neural network for ctr prediction. *arXiv preprint arXiv:1703.04247*, 2017. (Cited on page 21.)

[168] Guolin Ke, Zhenhui Xu, Jia Zhang, Jiang Bian, and Tie-Yan Liu. Deepgbm: A deep learning framework distilled by gbdt for online prediction tasks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 384–394, 2019. (Cited on pages 21, 24, and 43.)

[169] Sergei Popov, Stanislav Morozov, and Artem Babenko. Neural oblivious decision ensembles for deep learning on tabular data. *arxiv:1909.06312*, 2019. (Cited on page 21.)

[170] Rishabh Agarwal, Nicholas Frosst, Xuezhou Zhang, Rich Caruana, and Geoffrey E Hinton. Neural additive models: Interpretable machine learning with neural nets. *arXiv preprint arXiv:2004.13912*, 2020. (Cited on page 21.)

[171] Liran Katzir, Gal Elidan, and Ran El-Yaniv. Net-DNF: Effective deep modeling of tabular data. In *International Conference on Learning Representations*, 2021. (Cited on page 21.)

[172] Ira Shavitt and Eran Segal. Regularization learning networks: deep learning for tabular datasets. In *Advances in Neural Information Processing Systems*, pages 1379–1389, 2018. (Cited on pages 21 and 42.)

[173] Yutaro Yamada, Ofir Lindenbaum, Sahand Negahban, and Yuval Kluger. Feature selection using stochastic gates. In *Proceedings of Machine Learning and Systems 2020*, pages 8952–8963. 2020. (Cited on page 21.)

[174] FICO. Home equity line of credit (HELOC) dataset. `https://community.fico.com/s/explainable-machine-learning-challenge`, 2019 (accessed June 15, 2022). (Cited on pages 21 and 44.)

[175] Pierre Baldi, Peter Sadowski, and Daniel Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nature communications*, 5(1):1–9, 2014. (Cited on page 21.)

[176] R Kelley Pace and Ronald Barry. Sparse spatial autoregressions. *Statistics & Probability Letters*, 33(3):291–297, 1997. (Cited on pages 21, 28, 29, 30, and 43.)

[177] Dirk Merkel. Docker: lightweight linux containers for consistent development and deployment. *Linux journal*, 2014(239):2, 2014. (Cited on pages 21 and 42.)

[178] Gang Luo. A review of automatic selection methods for machine learning algorithms and hyper-parameter values. *Network Modeling Analysis in Health Informatics and Bioinformatics*, 5(1):1–16, 2016. (Cited on page 22.)

[179] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019. (Cited on page 22.)

[180] Deandra Aulia Rusdah and Hendri Murfi. Xgboost in handling missing values for life insurance risk prediction. *SN Applied Sciences*, 2(8):1–10, 2020. (Cited on page 24.)

[181] Mustafa Alabadla, Fatimah Sidi, Iskandar Ishak, Hamidah Ibrahim, Lilly Suriani Affendey, Zafienas Che Ani, Marzanah A Jabar, Umar Ali Bukar, Navin Kumar Devaraj, Ahmad Sobri Muda, et al. Systematic review of using machine learning in imputing missing values. *IEEE Access*, 2022. (Cited on page 24.)

[182] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, et al. Practical lessons from predicting clicks on ads at facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*, pages 1–9, 2014. (Cited on pages 24 and 43.)

[183] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine learning*, 63(1):3–42, 2006. (Cited on page 24.)

[184] Frank Moosmann, Bill Triggs, and Frederic Jurie. Fast discriminative visual codebooks using randomized clustering forests. In *Twentieth Annual Conference on Neural Information Processing Systems (NIPS'06)*, pages 985–992. MIT Press, 2006. (Cited on page 24.)

[185] Xiaokang Qiu, Yuan Zuo, and Guannan Liu. Etcf: An ensemble model for ctr prediction. In *2018 15th International Conference on Service Systems and Service Management (ICSSSM)*, pages 1–5. IEEE, 2018. (Cited on page 24.)

[186] Zhabiz Gharibshah and Xingquan Zhu. User response prediction in online advertising. *aCM Computing Surveys (CSUR)*, 54(3):1–43, 2021. (Cited on page 25.)

[187] Jesús Bobadilla, Fernando Ortega, Antonio Hernando, and Abraham Gutiérrez. Recommender systems survey. *Knowledge-based systems*, 46:109–132, 2013. (Cited on page 25.)

[188] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. Multimodal deep learning. In *ICML*, 2011. (Cited on page 25.)

[189] Said Yacine Boulahia, Abdenour Amamra, Mohamed Ridha Madi, and Said Daikh. Early, intermediate and late fusion strategies for robust deep learning-based multimodal action recognition. *Machine Vision and Applications*, 32(6):1–18, 2021. (Cited on page 25.)

[190] Khaled Bayoudh, Raja Knani, Fayçal Hamdaoui, and Abdellatif Mtibaa. A survey on deep multimodal learning for computer vision: advances, trends, applications, and datasets. *The Visual Computer*, 38(8):2939–2970, 2022. (Cited on page 25.)

[191] Daniele Di Mitri, Jan Schneider, Marcus Specht, and Hendrik Drachsler. From signals to knowledge: A conceptual model for multimodal learning analytics. *Journal of Computer Assisted Learning*, 34(4):338–349, 2018. (Cited on page 25.)

[192] David Lichtenwalter, Peter Burggräf, Johannes Wagner, and Tim Weißer. Deep multimodal learning for manufacturing problem solving. *Procedia CIRP*, 99:615–620, 2021. (Cited on page 25.)

[193] Xingjian Shi, Jonas Mueller, Nick Erickson, Mu Li, and Alex Smola. Multimodal AutoML on structured tables with text fields. In *8th ICML Workshop on Automated Machine Learning (AutoML)*, 2021. (Cited on page 25.)

[194] Sebastian Pölsterl, Tom Nuno Wolf, and Christian Wachinger. Combining 3d image and tabular data via the dynamic affine feature map transform. *arXiv preprint arXiv:2107.05990*, 2021. (Cited on page 25.)

[195] Kuan Liu, Yanen Li, Ning Xu, and Prem Natarajan. Learn to combine modalities in multimodal deep learning. *arXiv preprint arXiv:1805.11730*, 2018. (Cited on page 25.)

[196] N. Patki, R. Wedge, and K. Veeramachaneni. The synthetic data vault. In *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 399–410, Oct 2016. (Cited on page 29.)

[197] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018. (Cited on page 28.)

[198] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019. (Cited on page 28.)

[199] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems (NeurIPS)*, 33:1877–1901, 2020. (Cited on page 28.)

[200] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. (Cited on page 30.)

[201] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. *CoRR*, abs/1703.01365, 2017. (Cited on page 33.)

[202] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net, 2015. (Cited on page 33.)

[203] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017. (Cited on page 33.)

[204] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. (Cited on pages 33 and 42.)

[205] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998. (Cited on pages 33 and 42.)

[206] Naman Bansal, Chirag Agarwal, and Anh Nguyen. Sam: The sensitivity of attribution methods to hyperparameters. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pages 8673–8683, 2020. (Cited on page 32.)

[207] Chun-Xia Zhang, Jiang-She Zhang, Nan-Nan Ji, and Gao Guo. Learning ensemble classifiers via restricted boltzmann machines. *Pattern Recognition Letters*, 36:161–170, 2014. (Cited on page 34.)

[208] Zhi-Hua Zhou. *Ensemble methods: foundations and algorithms*. Chapman and Hall/CRC, 2019. (Cited on page 34.)

[209] Omer Sagi and Lior Rokach. Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1249, 2018. (Cited on page 34.)

[210] Mudasir A Ganaie, Minghui Hu, et al. Ensemble deep learning: A review. *arXiv preprint arXiv:2104.02395*, 2021. (Cited on page 34.)

[211] Ludmila I Kuncheva and Christopher J Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, 51(2):181–207, 2003. (Cited on page 34.)

[212] Ariel Jaffe, Ethan Fetaya, Boaz Nadler, Tingting Jiang, and Yuval Kluger. Unsupervised ensemble learning with dependent classifiers. In *Artificial Intelligence and Statistics*, pages 351–360. PMLR, 2016. (Cited on page 34.)

[213] Geoffrey E Hinton. A practical guide to training restricted boltzmann machines. In *Neural networks: Tricks of the trade*, pages 599–619. Springer, 2012. (Cited on page 34.)

[214] Uri Shaham, Xiuyuan Cheng, Omer Dror, Ariel Jaffe, Boaz Nadler, Joseph Chang, and Yuval Kluger. A deep learning approach to unsupervised ensemble learning. In *International conference on machine learning*, pages 30–39. PMLR, 2016. (Cited on page 34.)

[215] Klaus Broelemann, Thomas Gottron, and Gjergji Kasneci. Restricted boltzmann machines for robust and fast latent truth discovery. *arXiv preprint arXiv:1801.00283*, 2017. (Cited on page 34.)

[216] Alexander Philip Dawid and Allan M Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 28(1):20–28, 1979. (Cited on page 34.)

[217] Gjergji Kasneci, Jurgen Van Gael, Ralf Herbrich, and Thore Graepel. Bayesian knowledge corroboration with logical rules and user feedback. In *Joint european conference on machine learning and knowledge discovery in databases*, pages 1–18. Springer, 2010. (Cited on page 34.)

[218] Gjergji Kasneci, Jurgen Van Gael, David Stern, and Thore Graepel. Cobayes: bayesian knowledge corroboration with assessors of unknown areas of expertise. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 465–474, 2011. (Cited on page 34.)

[219] Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. A benchmark for interpretability methods in deep neural networks. *Advances in neural information processing systems*, 32, 2019. (Cited on pages 35, 36, and 45.)

[220] Henry J Kelley. Gradient theory of optimal flight paths. *Ars Journal*, 30(10):947–954, 1960. (Cited on page 37.)

[221] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *arXiv:1912.01703*, 2019. (Cited on page 37.)

[222] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems. 2015. Software available from tensorflow.org. (Cited on page 37.)

[223] François Chollet et al. Keras. `https://keras.io`, 2015. (Cited on page 37.)

[224] Wei-Chao Lin, Chih-Fong Tsai, and Jia Rong Zhong. Deep learning for missing value imputation of continuous data and the effect of data discretization. *Knowledge-Based Systems*, 239:108079, 2022. (Cited on page 42.)

[225] Cheng Fan, Meiling Chen, Xinghua Wang, Jiayuan Wang, and Bufu Huang. A review on data preprocessing techniques toward efficient and reliable knowledge discovery from building operational data. *Frontiers in Energy Research*, 9:652801, 2021. (Cited on page 42.)

[226] Salvador García, Julián Luengo, and Francisco Herrera. *Data preprocessing in data mining*, volume 72. Springer, 2015. (Cited on page 42.)

[227] Besim Bilalli et al. Learning the impact of data pre-processing in data analysis. 2018. (Cited on page 42.)

[228] Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. TaBERT: Pretraining for joint understanding of textual and tabular data. *arxiv:2005.08314*, 2020. (Cited on page 42.)

[229] Yaru Hao, Li Dong, Furu Wei, and Ke Xu. Self-attention attribution: Interpreting information interactions inside transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12963–12971, 2021. (Cited on page 42.)

[230] Sarthak Jain and Byron C Wallace. Attention is not explanation. *arXiv preprint arXiv:1902.10186*, 2019. (Cited on page 42.)

[231] Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. *arXiv preprint arXiv:1908.04626*, 2019. (Cited on page 42.)

[232] Gilmer Valdes, Wilmer Arbelo, Yannet Interian, and Jerome H Friedman. Lockout: Sparse regularization of neural networks. *arXiv preprint arXiv:2107.07160*, 2021. (Cited on page 42.)

[233] Joseph Chee Chang, Saleema Amershi, and Ece Kamar. Revolt: Collaborative crowdsourcing for labeling machine learning datasets. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 2334–2346, 2017. (Cited on page 42.)

[234] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge and Data Engineering*, 2021. (Cited on page 43.)

[235] Dara Bahri, Heinrich Jiang, Yi Tay, and Donald Metzler. SCARF: Self-supervised contrastive learning using random feature corruption. *arXiv preprint arXiv:2106.15147*, 2021. (Cited on page 43.)

[236] Talip Ucar, Ehsan Hajiramezanali, and Lindsay Edwards. Subtab: Subsetting features of tabular data for self-supervised representation learning. *Advances in Neural Information Processing Systems*, 34, 2021. (Cited on page 43.)

[237] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 eighth ieee international conference on data mining*, pages 413–422. IEEE, 2008. (Cited on page 43.)

[238] Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. Do NLP models know numbers? probing numeracy in embeddings. In *Empirical Methods in Natural Language Processing*, 2019. (Cited on page 44.)

[239] Mor Geva, Ankit Gupta, and Jonathan Berant. Injecting numerical reasoning skills into language models. *arXiv preprint arXiv:2004.04487*, 2020. (Cited on page 44.)

[240] Max Kuhn and Kjell Johnson. *Feature engineering and selection: A practical approach for predictive models*. CRC Press, 2019. (Cited on page 44.)

[241] Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910*, 2022. (Cited on page 44.)

[242] Eirini Ntoutsi, Pavlos Fafalios, Ujwal Gadiraju, Vasileios Iosifidis, Wolfgang Nejdl, Maria-Esther Vidal, Salvatore Ruggieri, Franco Turini, Symeon Papadopoulos, Emmanouil Krasanakis, et al. Bias in data-driven artificial intelligence sys-

tems—an introductory survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 10(3):e1356, 2020. (Cited on page 45.)

[243] Amit Giloni, Edita Grolman, Tanja Hagemann, Ronald Fromm, Sebastian Fischer, Yuval Elovici, and Asaf Shabtai. BENN: Bias estimation using a deep neural network. *IEEE Transactions on Neural Networks and Learning Systems*, 2022. (Cited on page 45.)

[244] A. Datta, S. Sen, and Y. Zick. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 2016. (Cited on page 46.)

[245] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. (Cited on page 47.)

[246] Angela Fan, Thibaut Lavril, Edouard Grave, Armand Joulin, and Sainbayar Sukhbaatar. Addressing some limitations of transformers with feedback memory. *arXiv preprint arXiv:2002.09402*, 2020. (Cited on page 47.)

[247] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016. (Cited on page 47.)

[248] Roman Levin, Valeriia Cherepanova, Avi Schwarzschild, Arpit Bansal, C Bayan Bruss, Tom Goldstein, Andrew Gordon Wilson, and Micah Goldblum. Transfer learning with deep tabular models. *arXiv preprint arXiv:2206.15306*, 2022. (Cited on page 47.)

[249] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. Fooling lime and shap: Adversarial attacks on post hoc explanation methods. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 180–186, 2020. (Cited on page 47.)

[250] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019. (Cited on page 48.)

[251] Alexandra Zytek, Dongyu Liu, Rhema Vaithianathan, and Kalyan Veeramachaneni. Sibyl: Understanding and addressing the usability challenges of machine learning in high-stakes decision making. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):1161–1171, 2021. (Cited on page 48.)

[252] Alan M Turing. Computing machinery and intelligence. In *Parsing the turing test*, pages 23–65. Springer, 2009. (Cited on page 48.)

Appendix A

Appendix

This thesis is based on nine manuscripts, all of which are publicly available online. The versions presented here are identical to those published online.

I am the first author or the primary contributor on the majority of these manuscripts. Please refer to the subsequent sections for a more detailed breakdown of each author's contributions.

A.1 Deep Tabular Data Learning Publications

A.1.1 Deep Neural Networks and Tabular Data: A Survey

Publication: Published in the IEEE Transactions on Neural Networks and Learning Systems journal, 2022.

Contribution: I came up with an idea about the survey of the deep neural networks and heterogeneous tabular data for several topics – inference, synthetic data generation, and explainability. Kathrin Seßler and I collected and analyzed methods for the inference tasks. Tobias Leemann collected and analyzed approaches for tabular generation tasks. Martin Pawelczyk analyzed the explainability of deep neural networks for tabular data in the literature. Together, Kathrin Seßler, Tobias Leemann, and I developed a framework for evaluating machine learning methods on tabular data. Gjergji Kasneci provided valuable input on the structure of the survey. All co-authors and I contributed to the revision of the final manuscript.

Deep Neural Networks and Tabular Data: A Survey

Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci

***Abstract*— Heterogeneous tabular data are the most commonly used form of data and are essential for numerous critical and computationally demanding applications. On homogeneous datasets, deep neural networks have repeatedly shown excellent performance and have therefore been widely adopted. However, their adaptation to tabular data for inference or data generation tasks remains highly challenging. To facilitate further progress in the field, this work provides an overview of state-of-the-art deep learning methods for tabular data. We categorize these methods into three groups: data transformations, specialized architectures, and regularization models. For each of these groups, our work offers a comprehensive overview of the main approaches. Moreover, we discuss deep learning approaches for generating tabular data and also provide an overview over strategies for explaining deep models on tabular data. Thus, our first contribution is to address the main research streams and existing methodologies in the mentioned areas while highlighting relevant challenges and open research questions. Our second contribution is to provide an empirical comparison of traditional machine learning methods with 11 deep learning approaches across five popular real-world tabular datasets of different sizes and with different learning objectives. Our results, which we have made publicly available as competitive benchmarks, indicate that algorithms based on gradient-boosted tree ensembles still mostly outperform deep learning models on supervised learning tasks, suggesting that the research progress on competitive deep learning models for tabular data is stagnating. To the best of our knowledge, this is the first in-depth overview of deep learning approaches for tabular data; as such, this work can serve as a valuable starting point to guide researchers and practitioners interested in deep learning with tabular data.**

***Index Terms*— Benchmark, deep neural networks, discrete data, heterogeneous data, interpretability, probabilistic modeling, survey, tabular data, tabular data generation.**

I. Introduction

EVER-INCREASING computational resources and the availability of large, labeled datasets have accelerated the success of deep neural networks [1], [2]. In particular, architectures based on convolutions, recurrent mechanisms [3], [4], or transformers [5] have led to unprecedented performance in a multitude of domains. Although deep learning methods perform outstandingly well for classification or data generation tasks on homogeneous data (e.g., image, audio, and text data), tabular data still pose a challenge to deep learning models [6], [7], [8]. Tabular data—in contrast to image or language data—are heterogeneous, leading to dense numerical and sparse categorical features. Furthermore, the correlation among the features is weaker than the one introduced through spatial or semantic relationships in image or speech data. Hence, it is necessary to discover and exploit relations without relying on spatial information [9]. Therefore, Kadra et al. [10] called tabular datasets the "last unconquered castle" for deep neural network models.

Heterogeneous data are the most commonly used form of data [8], and it is ubiquitous in many crucial applications, such as medical diagnosis based on patient history [11], [12], [13], predictive analytics for financial applications (e.g., risk analysis, estimation of creditworthiness, the recommendation of investment strategies, and portfolio management) [14], click-through rate (CTR) prediction [15], user recommendation systems [16], [17], customer churn prediction [18], cybersecurity [19], fraud detection [20], psychology [21], anomaly detection [22], [23], [24], and so forth. In all these applications, a boost in predictive performance and robustness may have considerable benefits for both end users and companies that provide such solutions. Simultaneously, this requires handling many data-related pitfalls, such as noise, impreciseness, different attribute types and value ranges, or the missing value problem and privacy issues.

Meanwhile, deep neural networks offer multiple advantages over traditional machine learning methods. First, these methods are highly flexible [25], allow for efficient and iterative training, and are particularly valuable for AutoML [26], [27]. Second, tabular data generation is possible using deep neural networks and can, for instance, help mitigate class imbalance problems [28]. Third, neural networks can be deployed for multimodal learning problems where tabular data can be one of many input modalities [29], [30], for tabular data distillation [31], [32], for federated learning [33], and in many more scenarios.

Successful deployments of data-driven applications require solving several tasks, among which we identified three core challenges: 1) inference; 2) data generation; and 3) interpretability. The most crucial task is inference, which is concerned with making predictions based on past observations. While a powerful predictive model is critical for all the applications mentioned in the previous paragraph, the interplay between tabular data and deep neural networks goes beyond simple inference tasks. Before a predictive model can even be trained, the training data usually need to be preprocessed. This is where data generation plays a crucial role, as one of the standard deployment steps involves the imputation of missing values [34], [35] and the rebalancing of the dataset [36], [37] (i.e., equalizing sample sizes for different classes). Furthermore, it might be simply impossible to use the actual data due to privacy concerns, e.g., in financial or medical

Manuscript received 21 February 2022; revised 29 June 2022, 24 October 2022, and 28 November 2022; accepted 12 December 2022. *(Corresponding authors: Vadim Borisov; Tobias Leemann.)*

The authors are with the Data Science and Analytics Research (DSAR) Group, University of Tübingen, 72070 Tübingen, Germany (e-mail: vadim.borisov@uni-tuebingen.de; tobias.leemann@uni-tuebingen.de).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2022.3229161.

Digital Object Identifier 10.1109/TNNLS.2022.3229161

applications [38], [39]. Thus, to tackle the data preprocessing and privacy challenges, probabilistic tabular data generation is essential. Finally, with stricter data protection laws such as California Consumer Privacy Act (CCPA) [40] and the European General Data Protection Regulation (EU GDPR) [41], which both mandate a right to explanations for automated decision systems (e.g., in the form or recourse [42]), interpretability is becoming a key aspect for predictive models used for tabular data [43], [44]. During deployment, interpretability methods also serve as a valuable tool for model debugging and auditing [45].

Evidently, apart from the core challenges of inference, generation, and interpretability, there are several other important subfields, such as working with data streams, distribution shifts, as well as privacy and fairness considerations that should not be neglected. Nevertheless, to navigate the vast body of literature, we focus on the identified core problems and thoroughly review the state of the art in this work. We will briefly discuss the remaining topics at the end of this survey.

Beyond reviewing current literature, we think that an exhaustive comparison between existing deep learning approaches for heterogeneous tabular data is necessary to put reported results into context. The variety of benchmarking datasets and the different setups often prevent the comparison of results across papers. In addition, important aspects of deep learning models, such as training and inference time, model size, and interpretability, are usually not discussed. We aim to bridge this gap by providing a comparison of the surveyed inference approaches with classical—yet very strong—baselines such as XGBoost [46]. We open-source our code, allowing researchers to reproduce and extend our findings.

In summary, the aims of this survey are to provide the following:

1) a thorough review of existing scientific literature on deep learning for tabular data;
2) a taxonomic categorization of the available approaches for classification and regression tasks on heterogeneous tabular data;
3) a presentation of the state of the art and promising paths toward tabular data generation;
4) an overview of existing explanation approaches for deep models for tabular data;
5) an extensive empirical comparison of traditional machine learning methods and deep learning models on multiple real-world heterogeneous tabular datasets;
6) a discussion on the main reasons for the limited success of deep learning on tabular data;
7) a list of open challenges related to deep learning for tabular data.

Accordingly, this survey is structured as follows. We discuss related works in Section II. To introduce the reader to the field, in Section III, we provide definitions of the key terms, a brief outline of the domain's history, and propose a unified taxonomy of current approaches to deep learning with tabular data. Section IV covers the main methods for modeling tabular data using deep neural networks. Section V presents an overview on tabular data generation using deep neural networks. An overview of explanation mechanisms for deep models for tabular data is presented in Section VI. In Section VII, we provide an extensive empirical comparison of machine and deep learning methods on real-world data, which also involves model size, runtime, and interpretability. In Section VIII, we summarize the state of the field and give future perspectives. Finally, we outline several open research questions before concluding in Section IX.

II. Related Work

To the best of our knowledge, there is no study dedicated exclusively to the application of deep neural networks to tabular data, spanning the areas of supervised and unsupervised learning, data synthesis, and interpretability. Prior works cover some of these aspects, but none of them systematically discusses the existing approaches in the broadness of this survey.

However, there are some works that cover parts of the domain. There is a comprehensive analysis of common approaches for categorical data encoding as a preprocessing step for deep neural networks by Hancock and Khoshgoftaar [47]. The authors compared existing methods for categorical data encoding on various tabular datasets and different deep learning architectures. We discuss the key categorical data encoding methods in Section IV-A1.

A recent survey by Sahakyan et al. [43] summarizes explanation techniques in the context of tabular data. Hence, we do not provide a detailed discussion of explainable machine learning for tabular data in this article. However, for the sake of completeness, we present some of the most relevant works in Section VI and highlight open challenges in this area.

Gorishniy et al. [48] empirically evaluated a large number of state-of-the-art deep learning approaches for tabular data on a wide range of datasets. He et al. [49] demonstrated that a tuned deep neural network model with a ResNet-like architecture shows comparable performance to some state-of-the-art deep learning approaches for tabular data.

Recently, Shwartz-Ziv and Armon [8] published a study on several different deep models for tabular data, including TabNet [6], NODE [7], and Net-DNF [50]. In addition, they compared deep learning approaches to gradient boosting decision tree (GBDT) algorithms regarding accuracy, training effort, inference efficiency, and hyperparameter optimization time. They observed that deep models had the best results on their chosen datasets, and however, not one single deep model could outperform all the others in general. The deep models were challenged by GBDTs, leading the authors to conclude that efficient tabular data modeling using deep neural networks is still an open research problem. In the face of this evidence, we aim to integrate the necessary background for future research on the inference problem and on the intertwined challenges of generation and explainability into a single work.

III. Tabular Data and Deep Neural Networks

A. Definitions

In this section, we give definitions for central terms used in this work. We also provide pointers to the original works for more detailed explanations of the methods.

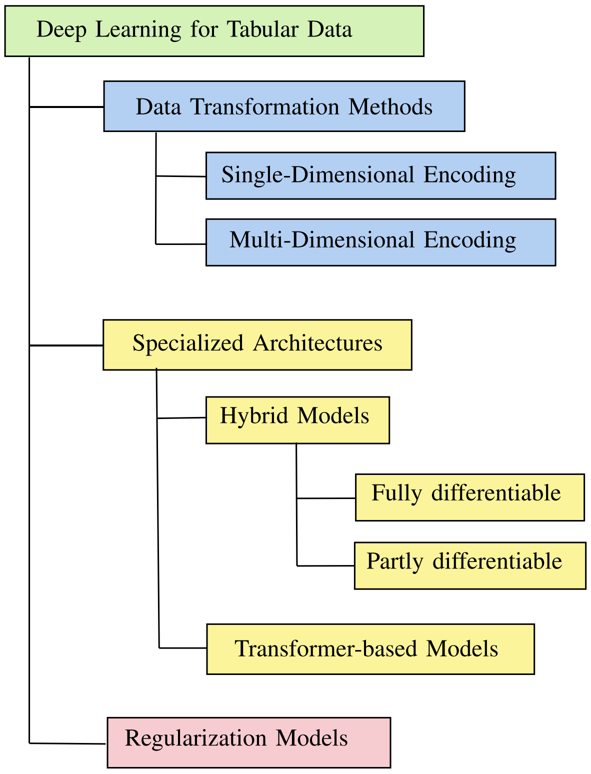

Deep Learning for Tabular Data
Data Transformation Methods
Single-Dimensional Encoding
Multi-Dimensional Encoding
Specialized Architectures
Hybrid Models
Fully differentiable
Partly differentiable
Transformer-based Models
Regularization Models

Fig. 1. Unified taxonomy of deep neural network models for heterogeneous tabular data.

The key concept in this survey is a (deep) neural network. Unless stated otherwise we use this concept as a synonym for feedforward networks, as described in [2], and name the concrete model whenever we deviate from this concept. A deep neural network defines mapping $\hat{f}$

$$y = f(x) \approx \hat{f}(x; W) \quad (1)$$

that learns the value of the model parameters W (i.e., the "weights" of a neural network) that results in the best approximation of the true underlying and unknown function f. In this case, x is a multidimensional data sample (i.e., $x \in \mathbb{R}^n$) with corresponding target y (where typically, $y \in \mathbb{R}^k$ for k classes and $y \in \mathbb{R}$ for regression tasks) from a dataset of tuples $\{(x_i, y_i)\}_{i \in \mathcal{I}}$. The network is called feedforward if the input information flows in one direction to the output without any feedback connections.

Throughout this survey, we focus on heterogeneous data that usually contain a variety of attribute types. These include both continuous and discrete attributes of different types (e.g., binary values, ordinal values, and high-cardinality categorical values). This is fundamentally different from homogeneous data modalities, such as images, audio, or text data where only a single feature type is present.

Categorical variables are an attribute type of particular importance. According to Lane's definition [51], categorical variables are qualitative values. They "do not imply a numerical ordering," unlike quantitative values, which are "measured in terms of numbers." Usually, a categorical variable can take one out of a limited set of values. Examples of typical categorical variables include gender, user_id, product_type and topic.

TABLE I

EXAMPLE OF A HETEROGENEOUS TABULAR DATASET. HERE, WE SHOW FIVE SAMPLES WITH SELECTED VARIABLES FROM THE ADULT DATASET [54]. SECTION VII-A PROVIDES FURTHER DETAILS ON THIS DATASET

Age	Education	Occupation	Sex	Income
39	Bachelors	Adm-clerical	Male	$\leq$50K
50	Bachelors	Exec-managerial	Male	$>$50K
38	HS-grad	Handlers-cleaners	Male	$\leq$50K
53	11th	Handlers-cleaners	Male	$\leq$50K
28	Bachelors	Prof-specialty	Female	$>$50K

Tabular data, sometimes also called structured data [52], are the subcategory of the heterogeneous data format that is usually presented in a table [53] with data points as rows and features as columns. In summary, for the scope of this work, we refer to a dataset with a fixed number of features that are either continuous or categorical as tabular. Each data point can be understood as a row in the table, or—taking a probabilistic view—as a sample from the unknown joint distribution. An illustrative example of five rows of heterogeneous, tabular data is provided in Table I.

B. Brief History of Deep Learning on Tabular Data

Tabular data are one of the oldest forms of data to be statistically analyzed. Before digital collection of text, images, and sound was possible, almost all data were tabular [55], [56], [57]. Therefore, it was the target of early machine learning research [58]. However, deep neural networks became popular in the digital age and were further developed with a focus on homogeneous data. In recent years, various supervised, self-supervised, and semisupervised deep learning approaches have been proposed, which explicitly address the issue of tabular data modeling again. Early works mostly focused on data transformation techniques for preprocessing [59], [60], which are still important today [47].

A huge stimulus was the rise of e-commerce, which demanded novel solutions, especially in advertising [15], [61]. These tasks required fast and accurate estimation on heterogeneous datasets with many categorical variables, for which the traditional machine learning approaches are not well suited (e.g., categorical features that have high cardinality can lead to very sparse high-dimensional feature vectors and nonrobust models). As a result, researchers and data scientists started looking for more flexible solutions, e.g., those based on deep neural networks, that can capture complex nonlinear dependencies in the data.

In particular, the CTR prediction problem has received a lot of attention [15], [62]. A large variety of approaches were proposed, most of them relying on specialized neural network architectures for heterogeneous tabular data.

A more recent line of research, sparked by Shavitt and Segal [63], evolved based on the idea that regularization may improve the performance of deep neural networks on tabular data [10]. This has led to an intensification of research on regularization approaches.

Due to the tremendous success of attention-based approaches such as transformers on textual [64] and visual data [65], [66], researchers have recently also started applying attention-based methods and self-supervised learning techniques to tabular data. After the introduction of transformer architectures to the field of tabular data [6], a lot of research effort has focused on transformer architectures that can be successfully applied to very large tabular datasets.

C. Challenges of Learning With Tabular Data

As we have mentioned in Section II, deep neural networks often perform less favorably compared to more traditional machine learning methods (e.g., tree-based methods) when dealing with tabular data. However, it is often unclear why deep learning cannot achieve the same level of predictive quality as in other domains such as image classification and natural language processing. In the following, we identify and discuss four possible reasons.

1) *Low-Quality Training Data:* Data quality is a common issue with real-world tabular datasets. They often include missing values [34], extreme data (outliers) [67], and erroneous or inconsistent data [68] and have a small overall size relative to the high-dimensional feature vectors generated from the data [69]. Also, due to the expensive nature of data collection, tabular data are frequently class-imbalanced. These challenges affect all machine learning algorithms; however, most of the modern decision tree-based algorithms can handle missing values or different/extreme variable ranges internally by looking for appropriate approximations and split values [46], [70], [71].
2) *Missing or Complex Irregular Spatial Dependencies:* There is often no spatial correlation between the variables in tabular datasets [72] or the dependencies between features are rather complex and irregular. When working with tabular data, the structure and relationships between its features have to be learned from scratch. Thus, the inductive biases used in popular models for homogeneous data, such as convolutional neural networks, are unsuitable for modeling this data type [50], [73], [74].
3) *Dependency on Preprocessing:* A key advantage of deep learning on homogeneous data is that it includes an implicit representation learning step [2], so only a minimal amount of preprocessing or explicit feature construction is required. However, for tabular data and deep neural networks, the performance may strongly depend on the selected preprocessing strategy [75]. Handling the categorical features remains particularly challenging [47] and can easily lead to a very sparse feature matrix (e.g., by using a one-hot encoding scheme) or introduce a synthetic ordering of previously unordered values (e.g., by using an ordinal encoding scheme). Finally, preprocessing methods for deep neural networks may lead to information loss, leading to a reduction in predictive performance [76].
4) *Importance of Single Features:* While typically changing the class of an image requires a coordinated change in many features, i.e., pixels, the smallest possible change of a categorical (or binary) feature can entirely flip a prediction on tabular data [63]. In contrast to deep neural networks, decision-tree algorithms can handle varying feature importance exceptionally well by selecting a single feature and appropriate threshold (i.e., splitting) values and "ignoring" the rest of the data sample. Shavitt and Segal [63] have argued that individual weight regularization may mitigate this challenge and motivated more work in this direction [10].

With these four fundamental challenges in mind, we continue by organizing and discussing the strategies developed to address them. We start by developing a suitable taxonomy.

D. Unified Taxonomy

In this section, we introduce a taxonomy of approaches that allows for a unified view of the field. We divide the works from the deep learning with tabular data literature into three main categories: data transformation methods, specialized architectures, and regularization models. In Fig. 1, we provide an overview of our taxonomy of deep learning methods for tabular data.

1) Data Transformation Methods: The methods in the first group transform categorical and numerical data. This is usually done to enable deep neural network models to better extract the information signal. Methods from this group do not require new architectures or adaptations of the existing data processing pipeline. Nevertheless, the transformation step comes at the cost of an increased preprocessing time. This might be an issue for high-load systems [77], particularly in the presence of categorical variables with high cardinality and growing dataset size. We can further subdivide this area into single-dimensional encodings and multidimensional encodings. The former encodings are employed to transform each feature independently while the latter encoding methods map an entire record to another representation.

2) Specialized Architectures: The biggest share of works investigates specialized architectures and suggests that a different deep neural network architecture is required for tabular data. Two types of architectures are of particular importance: hybrid models fuse classical machine learning approaches (e.g., decision trees) with neural networks, while transformer-based models rely on attention mechanisms.

3) Regularization Models: Finally, the group of regularization models claims that one of the main reasons for the moderate performance of deep learning models on tabular data is their extreme nonlinearity and model complexity. Therefore, strong regularization schemes are proposed as a solution. They are mainly implemented in the form of special-purpose loss functions.

We believe that our taxonomy may help practitioners find the methods of choice that can be easily integrated into their existing tool chain. For instance, applying data transformations can result in performance improvements while maintaining the current model architecture. Conversely, using specialized architectures, the data preprocessing pipeline can be kept intact.

IV. Deep Neural Networks for Tabular Data

In this section, we discuss the use of deep neural networks on tabular data for classification and regression tasks according to the taxonomy presented in Section III. We provide an overview of existing deep learning approaches in this area of research in Table II and examine the three methodological categories in detail: data transformation methods (see Section IV-A), architecture-based methods (see Section IV-B), and regularization-based models (see Section IV-C).

A. Data Transformation Methods

Most traditional approaches for deep neural networks on tabular data fall into this group. Interestingly, data preprocessing plays a relatively minor role in computer vision, even though the field is currently dominated by deep learning solutions [2]. There are many different possibilities to transform tabular data, and each may have a different impact on the learning results [47].

1) Single-Dimensional Encoding: One of the critical obstacles for deep learning with tabular data is categorical variables. Since neural networks only accept real number vectors as inputs, these values must be transformed before a model can use them. Therefore, the first class of methods attempts to encode categorical variables in a way suitable for deep learning models.

Approaches in this group [47] are divided into deterministic techniques, which can be used before training the model, and more complicated automatic techniques that are part of the model architecture. There are many ways for deterministic data encoding; hence, we restrict ourselves to the most common ones without the claim of completeness.

The simplest data encoding technique might be ordinal or label encoding. Every category is just mapped to a discrete numeric value, e.g., {`Apple`, `Banana`} are encoded as {0, 1}. One drawback of this method may be that it introduces an artificial order to previously unordered categories. Another straightforward method that does not induce any order is the one-hot encoding. One additional column for each unique category is added to the data. Only the column corresponding to the observed category is assigned the value one, with the other values being zero. In our example, `Apple` could be encoded as `(1,0)` and `Banana` as `(0,1)`. In the presence of a diverse set of categories in the data, this method can lead to high-dimensional sparse feature vectors and exacerbate the "curse of dimensionality" problem.

One approach that needs no extra columns and does not include any artificial order is the so-called leave-one-out encoding. It is based on the target encoding technique proposed in the work in [94], where every category is replaced with the mean of the target variable of that category. The leave-one-out encoding excludes the current row when computing the mean of the target variable to avoid overfitting. This approach is also used in the CatBoost framework [71], a state-of-the-art machine learning library for heterogeneous tabular data based on the gradient boosting algorithm [95].

A different strategy is hash-based encoding. Every category is transformed into a fixed-size value via a deterministic hash function. The output size is not directly dependent on the number of input categories but can be chosen manually.

2) Multidimensional Encoding: A first automatic encoding strategy is the value imputation and mask estimation (VIME) approach [79]. The authors propose a self-supervised and semisupervised deep learning framework for tabular data that trains an encoder in a self-supervised fashion by using two pretext tasks. Those tasks are independent of the concrete downstream task that the predictor has to solve. The first task of VIME is called mask vector estimation; its goal is to determine which values in a sample are corrupted. The second task, i.e., feature vector estimation, is to recover the original values of the sample. The encoder itself is a simple multilayer perceptron. This automatic encoding makes use of the fact that there is often much more unlabeled than labeled data. The encoder learns how to construct an informative homogeneous representation of the raw input data. In the semisupervised step, a predictive model, which is also a deep neural network model, is trained using the labeled and unlabeled data transformed by the encoder. For the encoder, a novel data augmentation method is used, corrupting an unlabeled data point multiple times with different masks. On the predictions from all augmented samples from one original data point, a consistency loss can be computed, which rewards similar outputs. To summarize, the VIME network trains an encoder, which is responsible to transform the categorical and numerical features into a new homogeneous and informative representation. This transformed feature vector is used as an input to the predictive model. For the encoder itself, the categorical data can be transformed by a simple one-hot encoding and binary encoding. The experimental results highlight how the self-supervised and semisupervised variants of the VIME framework can boost the performance over that of other baselines such as XGBoost. Even in the absence of unlabeled data, learning encodings in the proposed manner is shown to be beneficial for downstream performance.

Another stream of research aims at transforming the tabular input into a more homogeneous format. Since the revival of deep learning, convolutional neural networks have shown tremendous success in computer vision tasks. Therefore, Sun et al. [78] proposed the SuperTML method, which is a data conversion technique to transform tabular data into an image data format (2-D matrices), i.e., black-and-white images. On three datasets, SuperTML shows performance comparable with or superior to XGBoost.

The image generator for tabular data (IGTD) in [72] follows an idea similar to SuperTML. The IGTD framework converts tabular data into images to make use of classical convolutional architectures. As convolutional neural networks rely on spatial dependencies, the transformation into images is optimized by minimizing the difference between the feature distance ranking of the tabular data and the pixel distance ranking of the generated image. Every feature corresponds to one pixel,

TABLE II

Overview of Deep Learning Approaches for Tabular Data. We Organize Them in Categories Ordered Chronologically Inside the Groups. The "Interpretability" Column Indicates Whether the Approach Offers Some Form Interpretability for the Model's Decisions. The Key Characteristics of Every Model Are Summarized in the Last Column

	Method	Interpretability	Key Characteristics
Encoding	SuperTML [78]		Transform tabular data into images for CNNs
	VIME [79]		Self-supervised learning and contextual embedding
	IGTD [72]		Transform tabular data into images for CNNs
	SCARF [80]		Self-supervised contrastive learning
Architectures, Hybrid	Wide&Deep [81]		Embedding layer for categorical features
	DeepFM [15]		Factorization machine for categorical data
	SDT [82]	✓	Distill neural network into interpretable decision tree
	xDeepFM [83]		Compressed interaction network
	TabNN [84]		DNNs based on feature groups distilled from GBDT
	DeepGBM [62]		Two DNNs, distill knowledge from decision tree
	NODE [7]		Differentiable oblivious decision trees ensemble
	NAM [85]	✓	Separate neural networks for each input variable
	NON [86]		Network-on-network model
	DNN2LR [87]		Calculate cross feature wields with DNNs for LR
	Net-DNF [50]		Structure based on disjunctive normal form
	Boost-GNN [88]		GNN on top decision trees from the GBDT algorithm
	SDTR [89]		Hierarchical differentiable neural regression model
Architectures, Transformer	TabNet [6]	✓	Sequential attention structure
	TabTransformer [90]	✓	Transformer network for categorical data
	SAINT [9]	✓	Attention over both rows and columns
	ARM-Net [91]		Adaptive relational modeling with multi-headgated attention network
	Non-Param. Transformer [92]		Process the entire data set at once, use attention between data points
Regul.	RLN [63]	✓	Hyperparameters regularization scheme
	STG [93]		Stochastic gate regularization
	Regularized DNNs [10]		A "cocktail" of regularization techniques

which leads to compact images with similar features close at neighboring pixels. Thus, IGDTs can be used in the absence of domain knowledge. The authors show relatively solid results for data with strong feature relationships, but the method may fail if the features are independent or feature similarities cannot characterize the relationships. In their experiments, the authors used only gene expression profiles and molecular descriptors of drugs as data. This kind of data may lead to a favorable inductive bias, so the general viability of the approach remains unclear.

B. Specialized Architectures

Specialized architectures form the largest group of approaches for deep tabular data learning. In this group, the focus is on the development and investigation of novel deep neural network architectures designed specifically for heterogeneous tabular data. Guided by the types of available models, we divide this group into two subgroups: hybrid models (presented in IV-B1) and transformer-based models (discussed in IV-B2).

1) Hybrid Models: Most approaches for deep neural networks on tabular data are hybrid models. They transform the data and fuse successful classical machine learning approaches, often decision trees, with neural networks. We distinguish between fully differentiable models, which can be differentiated with respect to all their parameters and partly differentiable models.

a) Fully differentiable models: The fully differentiable models in this category offer a valuable property: They permit end-to-end deep learning for training and inference by means of gradient descent optimizers. Thus, they allow for highly efficient implementations in modern deep learning frameworks that exploit GPU or TPU acceleration throughout the code.

Popov et al. [7] proposed an ensemble of differentiable oblivious decision trees [96]—also known as the NODE framework for deep learning on tabular data. Oblivious decision trees use the same splitting function for all nodes on the same level and can therefore be easily parallelized. NODE is inspired by the successful CatBoost [71] framework. To make the whole architecture fully differentiable and benefit from end-to-end optimization, NODE utilizes the entmax transformation [97] and soft splits. In the original experiments, the NODE framework outperforms XGBoost and other GBDT models on many datasets. As NODE is based on decision tree ensembles, there is no preprocessing or transformation of the categorical data necessary. Decision trees are known to handle discrete features well. In the official implementation, strings are converted to integers using the leave-one-out encoding scheme. The NODE framework is widely used and provides a sound implementation that can be readily deployed.

Frosst and Hinton [82] contributed another model relying on soft decision trees (SDTs) to make neural networks more interpretable. They investigated training a deep neural network first, before using a mixture of its outputs and the ground-truth labels to train the SDT model in a second step. The authors showed that training a neural model first increases accuracy over SDTs that are directly learned from the data. However, their distilled trees still exhibit a performance gap to the neural networks that were fit in the initial step. Nevertheless, the model itself shows a clear relationship among different classes in a hierarchical fashion. It groups different categorical values based on the common patterns, e.g., digits 8 and 9 from the MNIST dataset [98]. To summarize, the proposed method allows for high interpretability and efficient inference, at the cost of slightly reduced accuracy.

Follow-up work [89] extends this line of research to heterogeneous tabular data and regression tasks and presents the SDT regressor (SDTR) framework. The SDTR is a neural network, which imitates a binary decision tree. Therefore, all neurons, such as nodes in a tree, get the same input from the data instead of the output from previous layers. In the case of deep networks, the SDTR could not beat other state-of-the-art models, but it has shown promising results in a low-memory setting, where single tree models and shallow architectures were compared.

Katzir et al. [50] followed the related idea. Their Net-DNF builds on the observation that every decision tree is merely a form of a Boolean formula, more precisely a disjunctive normal form. They use this inductive bias to design the architecture of a neural network, which is able to imitate the characteristics of the GBDT algorithm. The resulting Net-DNF was tested for classification tasks on datasets with no missing values, where it showed the results that are comparable to those of XGBoost [46]. However, the authors did not mention how to handle high-cardinality categorical data, as the used datasets contained mostly numerical and few binary features.

Linear models (e.g., linear and logistic regression) provide global interpretability but are inferior to complex deep neural networks. Usually, handcrafted feature engineering is required to improve the accuracy of linear models. Liu et al. [87] used a deep neural network to combine the features in a possibly nonlinear way; the resulting combination of features then serves as input to the linear model. In their approach—termed DDN2LR—this enhances the simple, interpretable linear model. In experimental evaluations, DNN2LR can outperform other more complex DNN models while maintaining some extent of interpretability.

The work by Cheng et al. [81] proposes a hybrid architecture that consists of linear and deep neural network models—Wide&Deep. A linear model that takes single features and a wide selection of handcrafted logical expressions on features as an input is enhanced by a deep neural network to improve the generalization capabilities. In addition, Wide&Deep learns an n-dimensional embedding vector for each categorical feature. All embeddings are concatenated resulting in a dense vector used as input to the neural network. The final prediction can be understood as a sum of both models. Experiments with a real-world system for app recommendation confirmed that users installed apps suggested by Wide&Deep were significantly more often than those provided by the previous model. A similar work by Guo and Berkhahn [99] proposes an embedding using deep neural networks for categorical variables.

Another contribution to the realm of Wide&Deep models is DeepFM [15]. The authors demonstrate that it is possible to replace the handcrafted feature transformations with learned factorization machines (FMs) [100]. The FM is an extension of a linear model designed to capture lower order interactions between features within high-dimensional and sparse data efficiently. Higher order interactions are modeled by a deep neural network. Similar to the original Wide&Deep model, DeepFM also relies on the same embedding vectors for its “wide” and “deep” parts. In contrast to the original Wide&Deep model, however, DeepFM alleviates the need for manual feature engineering. The experimental results show a solid improvement in CTR prediction tasks compared to a variety of models relying on either low- or high-order dependencies only and compared to other hybrid approaches.

Finally, network-on-network (NON) [86] is a classification model for tabular data, which focuses on capturing the intrafeature information efficiently. It consists of three components: a fieldwise network consisting of one unique deep neural network for every column to capture the column-specific information, an across-field network, which chooses the optimal operations based on the dataset, and an operation fusion network, connecting the chosen operations allowing for nonlinearities. As the optimal operations for the specific data are selected, the performance is considerably better than that of other deep learning models. However, decision trees, the current state-of-the-art models for tabular data, were not listed among the baselines. Also, training as many neural networks as columns and selecting the operations on the fly may lead to a long computation time.

b) Partly differentiable models: This subgroup of hybrid models aims at combining nondifferentiable approaches with deep neural networks. Models from this group usually utilize decision trees for the nondifferentiable part.

The DeepGBM model [62] combines the flexibility of deep neural networks with the preprocessing capabilities of GBDTs. DeepGBM consists of two neural networks—CatNN and GBDT2NN. While CatNN is specialized to handle sparse categorical features, GBDT2NN is specialized to deal with dense numerical features.

In the preprocessing step for the CatNN network, the categorical data are transformed via ordinal encoding (to convert the potential strings into integers), and the numerical features are discretized, as this network is specialized for categorical data. The GBDT2NN network distills the knowledge about the underlying dataset from a model based on GBDTs by accessing the leaf indices of the decision trees. This embedding based on decision tree leaves was first proposed in [101] for the random forest algorithm. Later, the same knowledge distillation strategy has been adopted for GBDTs [102].

Using the proposed combination of two deep neural networks, DeepGBM has a strong learning capacity for both categorical and numerical features. Distinctively, the authors implemented and tested DeepGBM's online prediction performance, which is significantly higher than that of GBDTs. On the downside, the leaf indices can be seen as meta categorical features since these numbers cannot be directly compared. Also, it is not clear how other data-related issues, such as missing values, different scaling of numeric features, and noise influence the predictions produced by the models.

The TabNN architecture, introduced by Ke et al. [84], is based on two principles: explicitly leveraging expressive feature combinations and reducing model complexity. It distills the knowledge from GBDTs to retrieve feature groups; it clusters them and then constructs the neural network based on those feature combinations. Also, structural knowledge from the trees is transferred to provide an effective initialization. The experimental results show that the performance of a GBDT model can be further improved by leveraging its feature sets in combination with neural encoders. Furthermore, TabNN shows promising results on streaming data. However, the construction of the network already takes different extensive computation steps of which one is only a heuristic to avoid an NP-hard problem. Unfortunately, these computational challenges and the unavailability of an implementation limit the practical usability of the network.

In similar spirit to DeepGBM and TabNN, the work by Ivanov and Prokhorenkova [88] proposed using GBDTs for the data prepossessing step. They exploited the fact that decision trees are special cases of directed graphs and process decision trees using graph neural networks. Thus, the proposed framework exploits the topology information from the decision trees using graph neural networks [103]. The resulting architecture is coined boosted graph neural network (BGNN). In multiple experiments, BGNN demonstrates that the proposed architecture is superior to other state-of-the-art graph neural networks in terms of predictive performance and training time and also outperforms GDBT models on most of the datasets.

2) Transformer-Based Models: Transformer-based approaches form another subgroup of model-based deep neural methods for tabular data. Inspired by the recent surge of interest in transformer-based methods and their successes on text and visual data [66], [104], researchers and practitioners have proposed multiple approaches using deep attention mechanisms [5] for heterogeneous tabular data.

TabNet [6] is one of the first transformer-based models for tabular data. Like a decision tree, the TabNet architecture comprises multiple subnetworks that are processed in a sequential hierarchical manner. According to [6], each subnetwork corresponds to one decision step. To train TabNet,

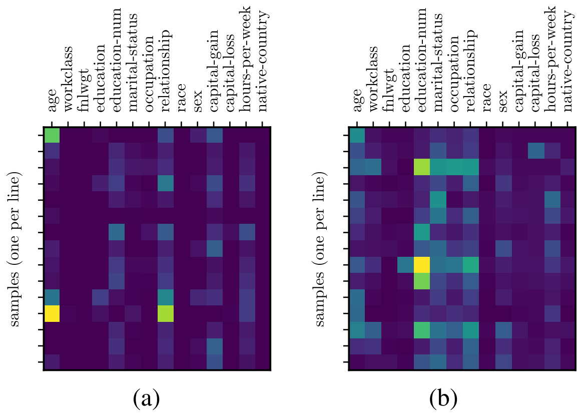


Fig. 2. Interpretable learning with the TabNet [6] architecture. We compare the attributions provided by the model for a sample from the UCI Adult dataset with those provided by the game theoretic KernelSHAP framework [116]. (a) TabNet attributions. (b) KernelSHAP attributions.

each decision step (subnetwork) receives the current data batch as input. TabNet aggregates the outputs of all decision steps to obtain the final prediction. At each decision step, TabNet first applies a sparse feature mask [105] to perform soft instancewise feature selection. The authors claim that the feature selection can save valuable resources, as the network may focus on the most important features. The feature mask of a decision step is trained using attentive information from the previous decision step. To this end, a feature transformer module decides which features should be passed to the next decision step and which features should be used to obtain the output at the current decision step. Some layers of the feature transformers are shared across all decision steps. The obtained feature masks correspond to local feature weights and can also be combined into a global importance score. Accordingly, TabNet is one of the few deep neural networks that offers different levels of interpretability by design. Indeed, experiments show that each decision step of TabNet tends to focus on a particular subdomain of the learning problem (i.e., one particular subset of features). This behavior is similar to convolutional neural networks. TabNet also provides a decoder module that is able to preprocess input data (e.g., replace missing values) in an unsupervised way. Accordingly, TabNet can be used in a two-stage self-supervised learning procedure, which improves the overall predictive quality. The experiments confirm the improved feature selection process, which leads to smaller models with less trainable parameters. Also, TabNet outperforms tree-based models and MLPs consistently while providing a more accurate interpretation of the feature importance. One of the popular Python [106] frameworks for tabular data provides an efficient implementation of TabNet [107]. Recently, TabNet has also been investigated in the context of fair machine learning [108], [109]. Attention-based architectures offer mechanisms for interpretability, which is an essential advantage over many hybrid models. Fig. 2 shows attention maps of the TabNet model and KernelSHAP explanation framework on the Adult dataset [54].

Another supervised and semisupervised approach is introduced by Huang et al. [90]. Their TabTransformer architecture uses self-attention-based transformers to map the categorical features to contextual embedding. This embedding is more robust to missing or noisy data and enables interpretability. The embedded categorical features are then together with the numerical ones fed into a simple multilayer perceptron. If, in addition, there is an extra amount of unlabeled data, unsupervised pretraining can improve the results, using masked language modeling or replacing token detection. Extensive experiments show that TabTransformer matches the performance of tree-based ensemble techniques, showing success also when dealing with missing or noisy data. The TabTransformer network puts a significant focus on the categorical features. It transforms the embedding of those features into contextual embedding, which is then used as input for the multilayer perceptron. This embedding is implemented by different multihead attention-based transformers, which are optimized during training.

ARM-net [91] is an adaptive neural network for relation modeling tailored to tabular data. The key idea of the ARM-net framework is to model feature interactions with combined features (feature crosses) selectively and dynamically by first transforming the input features into exponential space and then determining the interaction order and interaction weights adaptively for each feature cross. Furthermore, the authors propose a novel sparse attention mechanism to generate the interaction weights given the input data dynamically. Thus, users can explicitly model feature crosses of arbitrary orders with noisy features filtered selectively. On five real-world datasets, ARM-net shows its superior effectiveness in representing feature interactions compared to various baselines, which model the feature interactions in different ways.

Self-attention and intersample attention transformer (SAINT) [9] is a hybrid attention approach, combining self-attention [5] with intersample attention over multiple rows. When handling missing or noisy data, this mechanism allows the model to borrow the corresponding information from similar samples, which improves the model's robustness. The technique is reminiscent of nearest neighbor imputation. In addition, all features are embedded into a combined dense latent vector, enhancing existing correlations between values from one data point. To exploit the presence of unlabeled data, a self-supervised contrastive pre-training can further improve the results, minimizing the distance between two views of the same sample and maximizing the distance between different ones. Like the VIME framework (Section IV-A1), SAINT uses CutMix [110] to augment samples in the input space and uses mixup [111] in the embedding space. The experimental results show that SAINT outperforms tree-based models like XGBoost as well as other deep learning approaches for tabular data on average. When unlabeled data are available, the performance can be improved further using the proposed pretraining.

Finally, even some new learning paradigms are being proposed. For instance, the nonparametric transformer (NPT) [92] does not construct a mapping from individual inputs to outputs but uses the entire dataset at once. By using attention between data points, relations between arbitrary samples can be modeled and leveraged for classifying test samples. Experiments confirmed that this new approach can reach state-of-the-art results on most datasets by using intersample attention mechanisms.

C. Regularization Models

The third group of approaches argues that extreme flexibility of deep learning models for tabular data is one of the main learning obstacles and strong regularization of learned parameters may improve the overall performance.

One of the first methods in this category was the regularization learning network (RLN) proposed by Shavitt and Segal [63], which uses a learned regularization scheme. The main idea is based on the observation that features in tabular datasets have very different importances. Contrarily to other data modalities data such as images or text, a single tabular feature may change the entire prediction. Therefore, the authors apply trainable regularization coefficients to each single weight in a neural network, hence allowing high sensitivity with respect to certain inputs or network parts while being insensitive to others. To efficiently determine the corresponding coefficients, the authors propose a novel loss function termed "counterfactual loss." The regularization coefficients lead to a very sparse network, which also provides the importance of the remaining input features.

In their experiments, RLNs outperform deep neural networks and obtain the results comparable to those of the GBDT algorithm, but the evaluation relies on a dataset with mainly numerical data to compare the models. The RLN paper does not address the issues of categorical data. For the experiments and the example implementation, datasets with exclusively numerical data (except for the gender attribute) were used. A similar idea is proposed in [112], where regularization coefficients are learned only in the first layer with a goal to extract feature importance.

Kadra et al. [10] stated that simple multilayer perceptrons can outperform state-of-the-art algorithms on tabular data if deep learning networks are properly regularized. The authors propose a "cocktail" of regularization with 13 different techniques that are applied jointly. From those, the optimal subset and their subsidiary hyperparameters are selected. They demonstrate in extensive experiments that the regularization "cocktails" can not only improve the performance of multilayer perceptrons but these simple models also outperform tree-based architectures. On the downside, the extensive per-dataset regularization and hyperparameter optimization take much more computation time than the GBDT algorithm.

There are several other noteworthy works [113], [114], [115], indicating that strong regularization of deep neural networks can be beneficial for tabular data.

V. Tabular Data Generation

For many applications, the generation of realistic tabular data is fundamental. Three of the main purposes are data augmentation [117], data imputation (i.e., the filling of missing values) [118], [119], and rebalancing [36], [37], [120], [121]. Another highly relevant topic is privacy-aware machine learning [38], [39], [122] where generated data can potentially be leveraged to overcome privacy concerns.

A. Methods

While the generation of images and text is highly explored [123], [124], [125], generating synthetic tabular data is a less frequent concern. The mixed structure of discrete and continuous features along with their different value distributions still poses a significant challenge.

Classical approaches for the data generation task include Copulas [126], [127] and Bayesian networks [128]. Among Bayesian networks, those based on the Chow–Liu approximation [129] are especially popular [38], [130], [131], [132].

In the deep learning era, generative adversarial networks (GANs) [133] have proven highly successful for the generation of images [123], [134]. GANs were recently introduced as an original way to train a generative deep neural network model. They consist of two separate models: a generator G that generates samples from the data distribution and a discriminator D that estimates the probability that a sample came from the ground-truth distribution. Both G and D are usually chosen to be nonlinear functions such as multilayer perceptrons. To learn a generator distribution p_g over data $\boldsymbol{x}$, the generator $G(\boldsymbol{z}; \theta_g)$ maps the samples from a noise distribution $p_z(\boldsymbol{z})$ (e.g., the Gaussian distribution) to the input data space. The discriminator $D(\boldsymbol{x}; \theta_d)$ outputs the probability that a data point $\boldsymbol{x}$ comes from the training data's distribution p_{data} rather than from the generator's output distribution p_g. During joint training of G and D, G will start generating successively more realistic samples to fool the discriminator D. For more details on GANs, we refer the interested reader to the original paper [133].

In Table III, we provide an overview of tabular generation approaches that use deep learning techniques. Note that due to the enormous number of approaches, we list the most influential works that address the problem of data generation with a particular focus on tabular data. We exclude works that are targeted toward highly domain-specific tasks.

Although it was found that GANs lag behind at the generation of discrete outputs such as natural language [125], they are still frequently chosen to generate tabular data. Vanilla GANs or derivates, such as the Wasserstein GAN (WGAN) [135], WGAN with gradient penalty (WGAN-GP) [136], Cramér GAN [137], or the Boundary seeking GAN [138], which is designed to model discrete data, are commonly used in the literature to generate tabular data (cf. Table III). Moreover, VeeGAN [139] is frequently used as a reference for tabular data generation [38], [130], [131]. Apart from GANs, autoencoder-based architectures—in particular those relying on variational autoencoders (VAEs) [140]—have been proposed [130], [141].

In the following, we will briefly discuss the most relevant approaches that helped shape the domain. For example, MedGAN [39] was one of the first works and provides a deep learning model to generate patient records. As all the features in their work are discrete, this model cannot be easily transferred to arbitrary tabular datasets. The table-GAN approach in [142] adapts the deep convolutional GAN for tabular data. Specifically, the features from one record are converted into a matrix so that they can be processed by convolutional filters of a convolutional neural network. However, it remains unclear to which extent the inductive bias used for images are suitable for tabular data.

The approach by Xu et al. [130] focuses on the correlation between the features of one data point. The authors first propose the mode-specific normalization technique for data preprocessing that allows to transform non-Gaussian distributions in the continuous columns. They express numeric values in terms of a mixture component number and the deviation from that component's center. This allows to represent multimodal and skewed distributions. Their generative solution, coined CTGAN, uses the conditional GAN architecture to enforce learning proper conditional distributions for each column. To obtain categorical values and to allow for backpropagation in the presence of categorical values, the gumbel-softmax trick [143] is utilized. The authors also propose a model based on VAEs, named tabular VAE (TVAE), which outperforms their suggested GAN approach. Both approaches can be considered state of the art.

While GANs and VAEs are prevalent, other recently proposed architectures include machine-learned causal models [144] and invertible flows [38]. When privacy is the main factor of concern, models, such as PATE-GAN [145], provide generative models with certain differential privacy guarantees. Although very relevant for practical applications, such privacy guarantees and related federated learning approaches with tabular data [146] are outside the scope of this review.

Fan et al. [122] compared a variety of different GAN architectures for tabular data synthesis and recommended using a simple, fully connected architecture with a vanilla GAN loss with minor changes to prevent mode collapse. They also use the normalization proposed in [130]. In their experiments, the WGAN loss or the use of convolutional architectures on tabular data does boost the generative performance.

B. Assessing Generative Quality

To assess the quality of the generated data, several performance measures are used. The most common approach is to define a proxy classification task and train one model for it on the real training set and another on the artificially generated dataset. With a highly capable generator, the predictive performance of the artificial-data model on the real-data test set should be almost on par with its real-data counterpart. This measure is often referred to as machine learning efficacy and used in [39], [131], and [147]. In nonobvious classification tasks, an arbitrary feature can be used as a label and predicted [39], [148], [149]. Another approach is to visually inspect the modeled distributions per feature, e.g., the cumulative distribution functions [117], or compare the expected values in scatter plots [39], [148]. A more quantitative approach is the use of statistical tests, such as the Kolmogorov–Smirnov test [152], to assess the distributional difference [149]. On synthetic datasets, the output distribution can be compared to the ground truth, e.g., in terms of log likelihood [130], [144]. Because overfitted models can also obtain good scores, Xu et al. [130] proposed evaluating the likelihood of a test set under an estimate of the GAN's output distribution. Especially in a privacy-preserving context,

TABLE III
GENERATION OF TABULAR DATA USING DEEP NEURAL NETWORK MODELS (IN CHRONOLOGICAL ORDER)

Method	Based upon	Application
medGAN, medWGAN [39]	Autoencoder+GAN	Medical Records
TableGAN [142]	DCGAN	General
Mottini et al. [147]	Cramér GAN	Passenger Records
Camino et al. [148]	medGAN, ARAE	General
medBGAN, medWGAN [149]	WGAN-GP, Boundary seeking GAN	Medical Records
ITS-GAN [117]	GAN with AE for constraints	General
CTGAN, TVAE [130]	Wasserstein GAN, VAE	General
artGAN [121]	WGAN-GP	Health Data
VAEM [141]	VAE (Hierarchical)	General
OVAE [131]	Oblivious VAE	General
TAEI [37]	AE+SMOTE (in multiple setups)	General
Causal-TGAN [150]	Causal-Model, WGAN-GP	General
Copula-Flow [38]	Invertible Flows	General
Synthsonic [132]	Copula + CLBNs	General
GReaT [151]	Language Transformer	General

the distribution of the distance to closest record (DCR) can be calculated and compared to the respective distances on the test set [142]. This measure is important to assess the extent of sample memorization. Overall, we conclude that a single measure is not sufficient to assess the generative quality. For instance, a generative model that memorizes the original samples will score well in the machine learning efficiency metric but fail the DCR check. Therefore, we highly recommend using several evaluation measures that focus on individual aspects of data quality.

VI. EXPLANATION MECHANISMS FOR DEEP LEARNING WITH TABULAR DATA

Explainable machine learning is concerned with the problem of providing explanations for complex machine learning models. With stricter regulations for automated decision-making [41] and the adoption of machine learning models in high-stakes domains such as finance and healthcare [45], [153], [154], interpretability is becoming a key concern. Toward this goal, various streams of research follow different explainability paradigms. Among these, feature attribution methods and counterfactual explanations are two of the popular forms [155], [156], [157]. Because these techniques are gaining importance for researchers and practitioners alike, we dedicate the following to reviewing these methods.

A. Feature Highlighting Explanations

Local input attribution techniques seek to explain the behavior of machine learning models instance by instance. Those methods aim to highlight the influence of the inputs that have on the prediction by assigning importance scores to the input features. Some popular approaches for model explanations aim at constructing classification models that are explainable by design [158], [159], [160]. This is often achieved by enforcing the deep neural network model to be locally linear. Moreover, if the model's parameters are known and can be accessed, then the explanation technique can use these parameters to generate the model explanation. For such settings, relevance-propagation-based methods, e.g., [161], [162], and gradient-based approaches, e.g., [163], [164], [165], have been suggested. In cases where the parameters of the neural network cannot be accessed, model-agnostic approaches can prove useful. This group of approaches seeks to explain a model's behavior locally by applying surrogate models [116], [166], [167], [168], [169], which are interpretable by design and are used to explain individual predictions of black-box machine learning models. In order to test the performance of these black-box explanations techniques, Liu et al. [170] suggested a python-based benchmarking library.

B. Counterfactual Explanations

From the perspective of algorithmic recourse, the main purpose of counterfactual explanations is to suggest constructive interventions to the input of a deep neural network so that the output changes to the advantage of an end user. In simple terms, a minimal change to the feature vector that will flip the classification outcome is computed and provided as an explanation. By emphasizing both the feature importance and the recommendation aspect, counterfactual explanation methods can be further divided into three different groups: works that assume that all features can be independently manipulated [171] and works that focus on manifold constraints to capture feature dependencies.

In the class of independence-based methods, where the input features of the predictive model are assumed to be independent, some approaches use combinatorial solvers to generate recourse in the presence of feasibility constraints [172], [173], [174], [175]. Another line of research deploys gradient-based optimization to find low-cost counterfactual explanations in the presence of feasibility and diversity constraints [176], [177]. The main problem with these approaches is that they abstract from input correlations.

To alleviate this problem and to suggest realistic-looking counterfactuals, researchers have suggested building recourse suggestions on generative models [178], [179], [180], [181], [182]. The main idea is to change the geometry of the intervention space to a lower dimensional latent space, which encodes different factors of variation while capturing input dependencies. To this end, these methods primarily use (tabular data) VAEs [140], [183]. In particular, Mahajan et al. [181] demonstrated how to encode various feasibility constraints into such models. However, an extensive comparison across this class of methods is still missing since it is difficult to measure how realistic the generated data are in the context of algorithmic recourse.

More recently, a few works have suggested to develop counterfactual explanations that are robust to model shifts and noise in the recourse implementations [184], [185], [186]. A comprehensive treatment on how to extend these lines of work to arbitrary high-cardinality categorical variables is still an open problem in the field.

For a more fine-grained overview over the literature on counterfactual explanations, we refer the interested reader to the most recent surveys [187], [188]. Finally, Pawelczyk et al. [157] implemented an open-source python library, which provides support for many of the aforementioned counterfactual explanation models.

VII. Experiments

Although several experimental studies have been published in recent years [8], [10], an exhaustive comparison between existing deep learning approaches for heterogeneous tabular data is still missing in the literature. For example, important aspects of deep learning models, such as training and inference time, model size, and interpretability, are not discussed.

To fill this gap, we present an extensive empirical comparison of machine and deep learning methods on real-world datasets with varying characteristics in this section. We discuss the dataset choice (VII-A), the results (VII-B), and present a comparison of the training and inference time for all the machine learning models considered in this survey (VII-C). We also discuss the size of deep learning models. Finally, to the best of our knowledge, we present the first comparison of explainable deep learning methods for tabular data (VII-D). We release the full source code of our experiments for maximum transparency.[1]

A. Datasets

In computer vision, there are many established datasets for the evaluation of new deep learning architectures such as MNIST [98], CIFAR [189], and ImageNet [190]. On the contrary, there are no established standard heterogeneous datasets. Carefully checking the works listed in Section IV, we identified over 100 different datasets with different characteristics in their respective experimental evaluation sections. We note that the small overlap between the mentioned works makes it hard to compare the results across these works in general. Therefore, in this work, we deliberately select datasets covering the entire range of characteristics, such as data domain (e.g., finance, e-commerce, geography, and physics), different types of target variables (classification and regression), varying number of categorical variables and continuous variables, and differing sample sizes (small to large). Furthermore, most of the selected datasets were previously featured in multiple studies.

The first dataset of our study is the Home Equity Line of Credit (HELOC) dataset provided by FICO [191]. This dataset consists of anonymized information from real homeowners who applied for home equity lines of credit. An HELOC is a line of credit typically offered by a bank as a percentage of home equity. The task consists of using the information about the applicant in their credit report to predict whether they will repay their HELOC account within a two-year period.

[1]Open benchmarking on tabular data for machine learning models: https://github.com/kathrinse/TabSurvey.

TABLE IV
Main Properties of the Real-World Heterogeneous Tabular Datasets Used in This Survey. We Also Indicate the Dataset Task, Where "Binary" Stands for Binary Classification and "Multi-Class" Represents Multiclass Classification

	HELOC	Adult Income	HIGGS	Covertype	California Housing
#Samples	9.871	32.561	11 M.	581.012	20.640
#Num. features	21	6	27	52	8
#Cat. features	2	8	1	2	0
Task	Binary	Binary	Binary	Multi-Class	Regression
#Classes	2	2	2	7	-

We further use the Adult Income dataset [54], which is among the most popular tabular datasets used in the surveyed work (five usages). It includes basic information about individuals such as age, gender, and education. The target variable is binary; it represents high and low income.

The largest tabular dataset in our study is HIGGS, which stems from particle physics. The task is to distinguish between signals with Higgs bosons (HIGGS) and a background process [192]. Monte Carlo simulations [193] were used to produce the data. In the first 21 columns (columns 2-22), the particle detectors in the accelerator measure kinematic properties. In the last seven columns, these properties are analyzed. In total, HIGGS includes 11 million rows. We also binarize the 21st variable into a categorical variable with three groups since DeepFM, DeepGBM, TabTransformer, and SAINT models require at least one categorical attribute, to benchmark the method's special functionality on large datasets.

The Covertype dataset [54] is multiclassification dataset, which holds cartographic information about land cells (e.g., elevation and slope). The goal is to predict which one out of seven forest cover types is present in the cell.

Finally, we utilize the California Housing dataset [194], which contains information about a number of properties. The prediction task (regression) is to estimate the price of the corresponding home.

The fundamental characteristics of the selected datasets are summarized in Table IV.

B. Open Performance Benchmark on Tabular Data

1) Hyperparameter Selection: In order to do a fair evaluation, we use the Optuna library [199] with 100 iterations for each model to tune hyperparameters. Each hyperparameter configuration was cross-validated with five folds. The hyperparameter ranges used are publicly available online along with our code. We laid out the search space based on the information given in the corresponding papers and recommendations from the framework's authors.

2) Data Preprocessing: We prepossessed the data in the same way for every machine learning model by applying zero-mean, unit-variance normalization to the numerical features and an ordinal encoding to the categorical ones using the

TABLE V

OPEN PERFORMANCE BENCHMARK RESULTS BASED ON (STRATIFIED) FIVEFOLD CROSS VALIDATION. WE USE THE SAME FOLD SPLITTING STRATEGY FOR EVERY DATASET. THE TOP RESULTS FOR EACH DATASET ARE IN **BOLD**, WE ALSO UNDERLINE THE SECOND-BEST RESULTS. THE MEAN AND STANDARD DEVIATION VALUES ARE REPORTED FOR EACH BASELINE MODEL. MISSING RESULTS INDICATE THAT THE CORRESPONDING MODEL COULD NOT BE APPLIED TO THE TASK TYPE (REGRESSION OR MULTICLASS CLASSIFICATION)

	Method	HELOC		Adult		HIGGS		Covertype		Cal. Housing
		Acc ↑	AUC ↑	Acc ↑	AUC ↑	Acc ↑	AUC ↑	Acc ↑	AUC ↑	MSE ↓
Machine Learning	Linear Model	73.0±0.0	80.1±0.1	82.5±0.2	85.4±0.2	64.1±0.0	68.4±0.0	72.4±0.0	92.8±0.0	0.528±0.008
	KNN [58]	72.2±0.0	79.0±0.1	83.2±0.2	87.5±0.2	62.3±0.1	67.1±0.0	70.2±0.1	90.1±0.2	0.421±0.009
	Decision Trees [195]	80.3±0.0	89.3±0.1	85.3±0.2	89.8±0.1	71.3±0.0	78.7±0.0	79.1±0.0	95.0±0.0	0.404±0.007
	Random Forest [196]	82.1±0.2	90.0±0.2	86.1±0.2	91.7±0.2	71.9±0.0	79.7±0.0	78.1±0.1	96.1±0.0	0.272±0.006
	XGBoost [46]	83.5±0.2	92.2±0.0	87.3±0.2	92.8±0.1	77.6±0.0	85.9±0.0	**97.3±0.0**	**99.9±0.0**	0.206±0.005
	LightGBM [70]	83.5±0.1	92.3±0.0	**87.4±0.2**	**92.9±0.1**	77.1±0.0	85.5±0.0	93.5±0.0	99.7±0.0	**0.195±0.005**
	CatBoost [71]	**83.6±0.3**	**92.4±0.1**	87.2±0.2	92.8±0.1	77.5±0.0	85.8±0.0	96.4±0.0	99.8±0.0	0.196±0.004
	Model Trees [197]	82.6±0.2	91.5±0.0	85.0±0.2	90.4±0.1	69.8±0.0	76.7±0.0	-	-	0.385±0.019
Deep Learning	MLP [198]	73.2±0.3	80.3±0.1	84.8±0.1	90.3±0.2	77.1±0.0	85.6±0.0	91.0±0.4	76.1±3.0	0.263±0.008
	VIME [79]	72.7±0.0	79.2±0.0	84.8±0.2	90.5±0.2	76.9±0.2	85.5±0.1	90.9±0.1	82.9±0.7	0.275±0.007
	DeepFM [15]	73.6±0.2	80.4±0.1	86.1±0.2	91.7±0.1	76.9±0.0	83.4±0.0	-	-	0.260±0.006
	DeepGBM [62]	78.0±0.4	84.1±0.1	84.6±0.3	90.8±0.1	74.5±0.0	83.0±0.0	-	-	0.856±0.065
	NODE [7]	79.8±0.2	87.5±0.2	85.6±0.3	91.1±0.2	76.9±0.1	85.4±0.1	89.9±0.1	98.7±0.0	0.276±0.005
	NAM [85]	73.3±0.1	80.7±0.3	83.4±0.1	86.6±0.1	53.9±0.6	55.0±1.2	-	-	0.725±0.022
	Net-DNF [50]	82.6±0.4	91.5±0.2	85.7±0.2	91.3±0.1	76.6±0.1	85.1±0.1	94.2±0.1	99.1±0.0	-
	TabNet [6]	81.0±0.1	90.0±0.1	85.4±0.2	91.1±0.1	76.5±1.3	84.9±1.4	93.1±0.2	99.4±0.0	0.346±0.007
	TabTransformer [90]	73.3±0.1	80.1±0.2	85.2±0.2	90.6±0.2	73.8±0.0	81.9±0.0	76.5±0.3	72.9±2.3	0.451±0.014
	SAINT [9]	82.1±0.3	90.7±0.2	86.1±0.3	91.6±0.2	**79.8±0.0**	**88.3±0.0**	96.3±0.1	99.8±0.0	0.226±0.004
	RLN [63]	73.2±0.4	80.1±0.4	81.0±1.6	75.9±8.2	71.8±0.2	79.4±0.2	77.2±1.5	92.0±0.9	0.348±0.013
	STG [93]	73.1±0.1	80.0±0.1	85.4±0.1	90.9±0.1	73.9±0.1	81.9±0.1	81.8±0.3	96.2±0.0	0.285±0.006

alphabetical order. According to Hancock and Khoshgoftaar [47], the chosen encoding strategy shows comparable performance to more advanced methods. The missing values were substituted with zeros for the linear regression and models based on pure neural networks since these methods cannot accept them otherwise. We explicitly specify categorical features for LightGBM, DeepFM, DeepGBM, TabNet, TabTransformer, and SAINT since these approaches provide special functionality dedicated to categorical values, e.g., learning an embedding of the categories. As we noted in Section III-C, the results of experiments may be affected by the data preprocessing.

3) Reproducibility and Extensibility: For maximum reproducibility, we run all experiments in a docker container [200]. We underline again that our full code is publicly released so that the experiments can be replicated. The mentioned datasets are also publicly available and can be used as a benchmark for novel methods. We would highly welcome contributed implementations of additional methods from the data science community.

4) Results: The results of our experiments are shown in Table V. They draw a different picture than many recent research papers may suggest: for all but the very large HIGGS dataset, the best scores are still obtained by boosted decision tree ensembles. XGBoost and CatBoost outperform all deep learning-based approaches on the small and medium datasets, the regression dataset, and the multiclass dataset. For the large-scale HIGGS, SAINT outperforms the classical machine learning approaches. This suggests that for very large tabular datasets with predominantly continuous features, modern neural network architectures may have an advantage over classical approaches after all. In general, however, our results are consistent with the inferior performance of deep learning techniques in comparison to approaches based on decision tree ensembles (such as GBDT) on tabular data that were observed in various Kaggle competitions [201].

Considering only deep learning approaches, we observe that SAINT provided competitive results across datasets. However, for the other models, the performance was highly dependent on the chosen dataset. DeepFM performed best (among the deep learning models) on the Adult dataset and second-best on the California Housing dataset, but returned only weak results on the HELOC dataset.

C. Run Time Comparison

We also analyze the training and inference time of the models in comparison to their performance. We plot the time–performance characteristic for the models in Figs. 3 and 4 for the Adult and the HIGGS dataset, respectively. While the training time of gradient boosting-based models is lower than that of most deep neural network-based methods, their inference time on the HIGGS dataset with 11 million samples is significantly higher: for XGBoost, the inference time amounts to 5995 s, whereas inference times for MLP and SAINT are 10.18 and 282 s, respectively. All

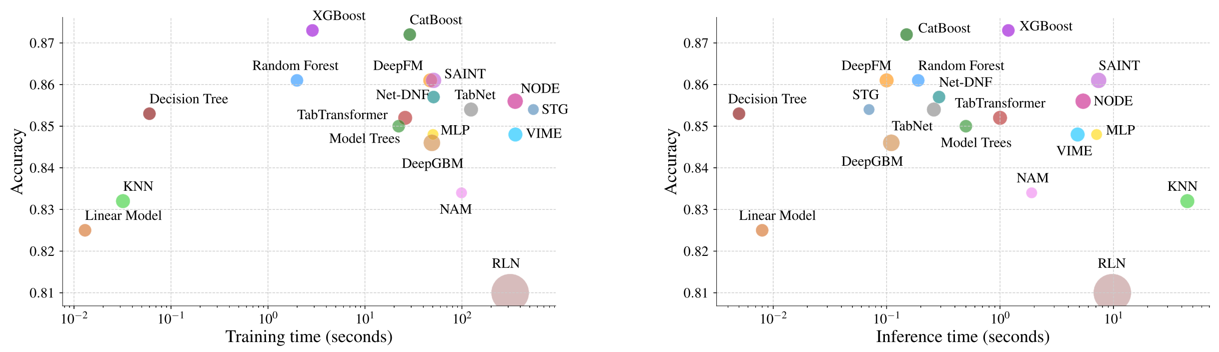


Fig. 3. Train (left) and inference (right) time benchmarks for selected methods on the Adult dataset with 32.561 samples. The circle size reflects the accuracy standard deviation.

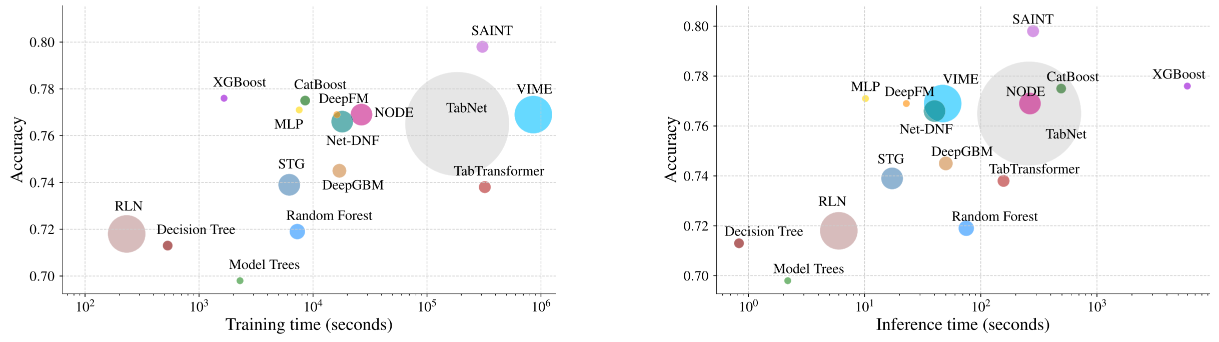


Fig. 4. Train (left) and inference (right) time benchmarks for selected methods on the HIGGS dataset with 11 million samples. The circle size reflects the accuracy standard deviation.

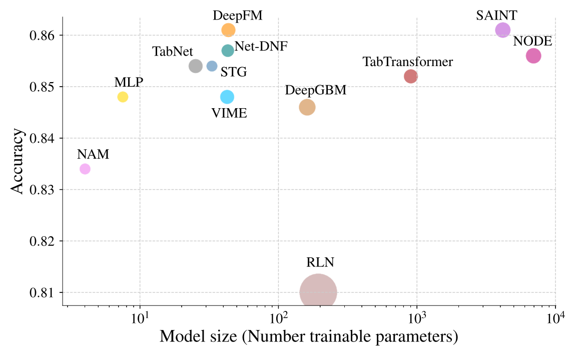


Fig. 5. Size comparison of deep learning models on the Adult dataset. The circle size reflects the standard deviation.

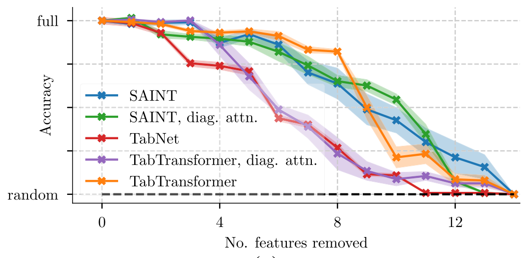
gradient boosting and deep learning models were trained on the same GPU.

D. Interpretability Assessment

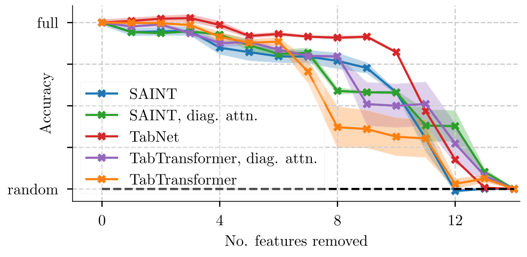
As opposed to the pure on-task performance, interpretability of the models is becoming an increasingly important characteristic. Therefore, we end this section with a distinct assessment of the interpretability properties claimed by some methods. The model size (number of parameters) can provide a first intuition of the interpretability of the models. Therefore, we provide a size comparison of deep learning models in Fig. 5.

Admittedly, explanations can be provided in very different forms, which may each have their own use cases. Hence, we can only compare explanations that have a common form. In this work, we chose feature attributions as the explanation format because they are the prevalent form of post hoc explainability for the models considered in this work. Remarkably, the models that build on the transformer architecture (Section IV-B2) often claim some extent of interpretability through the attention maps [9]. To verify this hypothesis and assess the attribution provided by some of the frameworks in practice, we run an ablation test with the features that were attributed the highest importance over all samples. Furthermore, due to the lack of ground-truth attribution values, we compare individual attributions to the well-known KernelSHAP values [116].

Evaluation of the quality of feature attribution is known to be a nontrivial problem [202]. We measure the fidelity [203] of the attributions by successively removing the features that have the highest mean importance assigned (most relevant first (MoRF) [203]). We then retrain the model on the reduced feature set. A sharp drop in predictive accuracy indicates that the discriminative features were successfully identified and removed. We do the same for the inverse order, least relevant first (LeRF), which removes the features deemed unimportant. In this case, the accuracy should stay high as long as possible. For the attention maps of TabTransformer and SAINT, we either use the sum over the entire columns of the intrafeature attention maps as an importance



(a)



(b)

Fig. 6. Resulting curves of the global attribution benchmark for feature attributions (fifteen runs on Adult). Standard errors are indicated by the shaded area. For the MoRF order, an early drop in accuracy is desirable, while for LeRF, the accuracy should stay as high as possible. (a) MoRF. (b) LeRF.

estimate or only take the diagonal (feature self-attentions) as attributions.

The obtained curves are visualized in Fig. 6. For the MoRF order, TabNet and TabTransformer with the diagonal of the attention head as attributions seem to perform best. For LeRF, TabNet is the only significantly better method than the others. For TabTransformer, taking the diagonal of the attention matrix seems to increase the performance, whereas for SAINT, there is almost no difference. We additionally compare the attribution values obtained to values from the KernelSHAP attribution method. Unfortunately, there are no ground-truth attributions to compare with. However, the SHAP framework has a solid grounding in game theory and is widely deployed [43]. We only compare the absolute values of the attributions, as the attention maps are constrained to be positive. As a measure of agreement, we compute the Spearman rank correlation between the attributions by the SHAP framework and the tabular data models and show the results in Table VI. The correlation we observe is surprisingly low across all models, and sometimes, it is even negative, which means that a higher SHAP attribution will probably result in a lower attribution by the model.

In these two simple benchmarks, the transformer models were not able to produce convincing feature attributions out-of-the-box. We come to the conclusion that more profound benchmarks of the claimed interpretability characteristics and their usefulness in practice are necessary.

TABLE VI

SPEARMAN RANK CORRELATION OF THE PROVIDED ATTRIBUTION WITH KERNELSHAP VALUES AS GROUND TRUTH. RESULTS WERE COMPUTED ON 750 RANDOM SAMPLES FROM THE ADULT DATASET

Model, attention used	Spearman Corr.
TabTransformer, columnw. attention	-0.01 $\pm$ 0.008
TabTransformer, diag. attention	0.00 $\pm$ 0.010
TabNet	**0.07** $\pm$ 0.009
SAINT, columnw. attention	-0.04 $\pm$ 0.007
SAINT, diag. attention	0.01 $\pm$ 0.007

VIII. DISCUSSION AND FUTURE PROSPECTS

In this section, we summarize our findings and discuss current and future trends in deep learning approaches for tabular data (Section VIII-A). Moreover, we identify several open research questions that could be tackled to advance the field of tabular deep neural networks (Section VIII-B).

A. Summary and Trends

1) Decision Tree Ensembles Are Still State of the Art: In a fair comparison on multiple datasets, we demonstrated that models based on tree ensembles, such as XGBoost, LightGBM, and CatBoost, still outperform the deep learning models on most datasets that we considered and come with the additional advantage of significantly less training time. Even though it has been six years since the XGBoost publication [46] and over 20 years since the publishing of original gradient boosting paper [95], we can state that despite much research effort in deep learning, the state of the art for tabular data remains largely unchanged. However, we observed that for very large datasets, approaches based on deep learning may still be able to achieve competitive performance and even outperform classical models. In summary, we think that a fundamental reorientation of the domain may be necessary. For now, the question of whether the use of current deep learning techniques is beneficial for tabular data can generally be answered in the negative. This applies in particular to small heterogeneous datasets that are common in applications. Hence, instead of proposing more and more complex models, we argue that a more profound understanding of the reasons for this performance gap is needed.

2) Unified Benchmarking: Furthermore, our results highlight the need for unified benchmarks. There is no consensus in the machine learning community on how to make a fair and efficient comparison. Shwartz-Ziv and Armon [8] showed that the choice of benchmarking datasets can have a non-negligible impact on the performance assessment. While we chose common datasets with varying characteristics for our experiments, a different choice of datasets or hyperparameter such as the encoding use (e.g., one-hot encoding for categorical variables) may lead to a different outcome. Because of the excessive number of datasets (in the 18 works listed in Table II, over 100 different datasets are used), there is a necessity for a standardized benchmarking procedure, which allows to identify significant progress with respect to the state of the art. With this work, we also propose an open-source benchmark for deep learning models on tabular data. For tabular data generation tasks, Xu et al. [130] proposed a sound evaluation framework with artificial and real-world datasets (Section V-B), but researchers need to agree on common benchmarks in this subdomain as well.

3) Tabular Data Preprocessing: Many of the challenges for deep neural networks on tabular data are related to the heterogeneity of the data (e.g., categorical and sparse values).

Therefore, some deep learning solutions transform them into a homogeneous representation more suitable to neural networks. While the additional overhead is small, such transforms can boost performance considerably and should thus be among the first strategies applied in real-world scenarios.

4) Architectures for Deep Learning on Tabular Data: Architecturewise, there has been a clear trend toward transformer-based solutions (Section IV-B2) in recent years. These approaches offer multiple advantages over standard neural network architectures, for instance, learning with attention over both categorical and numerical features. Moreover, self-supervised or unsupervised pretraining that leverages unlabeled tabular data to train parts of the deep learning model is gaining popularity, not only among transformer-based approaches. Performancewise, multiple independent evaluations demonstrate that deep neural network methods from the hybrid (Section IV-B1) and transformer-based (Section IV-B2) groups exhibit superior predictive performance compared to plain deep neural networks on various datasets [9], [48], [62], [84]. This underlines the importance of special-purpose architectures for tabular data.

5) Deep Generative Models for Tabular Data: Powerful tabular data generation is essential for the development of high-quality models, particularly in a privacy context. With suitable data generators at hand, developers can use large, synthetic, and yet realistic datasets to develop better models, while not being subject to privacy concerns [145]. Unfortunately, the generation task is as hard as inference in predictive models, so progress in both areas will likely go hand in hand.

6) Interpretable Deep Learning Models for Tabular Data: Interpretability is undoubtedly desirable, particularly for tabular data models frequently applied to personal data, e.g., in healthcare and finance. An increasing number of approaches offer it out-of-the-box, but most current deep neural network models are still mainly concerned with the optimization of a chosen error metric. Therefore, extending existing open-source libraries (see [157], [170]) aimed at interpreting black-box models helps advance the field. Moreover, interpretable deep tabular learning is essential for understanding model decisions and results, especially for life-critical applications. However, much of the state-of-the-art recourse literature does not offer easy support of heterogeneous tabular data and lacks metrics to evaluate the quality of heterogeneous data recourse. Finally, model explanations can be used to identify and mitigate potential unwanted biases and eliminate unfair discrimination [204], [205].

7) Learning From Evolving Data Streams: Many modern applications are subject to continuously evolving data streams, e.g., social media, online retail, or healthcare. Streaming data are usually heterogeneous and potentially unlimited. Therefore, observations must be processed in a single pass and cannot be stored. Indeed, online learning models can only access a fraction of the data at each time step. Furthermore, they have to deal with limited resources and shifting data distributions (i.e., concept drift) [206]. Hence, hyperparameter optimization and model selection, as typically involved in deep learning, are usually not feasible in a data stream. For this reason, despite the success of deep learning in other domains, less complex methods, such as incremental decision trees [207], [208], are often preferred in online learning applications.

B. Open Research Questions

Several open problems need to be addressed in future research. In this section, we will list those we deem fundamental to the domain.

1) Information-Theoretic Analysis of Encodings: Encoding methods are highly popular when dealing with tabular data. However, the majority of data preprocessing approaches for deep neural networks are lossy in terms of information content. Therefore, it is challenging to achieve an efficient, almost lossless transformation of heterogeneous tabular data into homogeneous data. Nevertheless, the information-theoretic view on these transformations remains to be investigated in detail and could shed light on the underlying mechanisms.

2) Computational Efficiency in Hybrid Models: The work by Shwartz-Ziv and Armon [8] suggests that the combination of a GBDT and deep neural networks may improve the predictive performance of a machine learning system. However, it also leads to growing complexity. Training or inference times, which far exceed those of classical machine learning approaches, are a recurring problem when developing hybrid models. We conclude that the integration of state-of-the-art approaches from classical machine learning and deep learning has not been conclusively resolved yet and future work should be conducted on how to mitigate the tradeoff between predictive performance and computational complexity.

3) Individual Regularizations: We applaud recent research on individual regularization methods, in which we see a promising direction to tackle the problem of highly sensitive features. We believe that representing the towering influence of certain features is crucial to success. Whether context- and architecture-specific regularizations for tabular data can be found remains an open question. In addition, it is relevant to explore the theoretical constraints that govern the success of regularization on tabular data more profoundly.

4) Novel Processes for Tabular Data Generation: For tabular data generation, modified GANs and VAEs are prevalent. However, the modeling of dependencies and categorical distributions remains the key challenge. Novel architectures in this area, such as diffusion models, have not been adapted to the domain of tabular data. Furthermore, the definition of an entirely new generative process particularly focused on tabular data might be worth investigating.

5) Interpretability: Going forward, counterfactual explanations for deep tabular learning can be used to improve the perceived fairness in human–artificial intelligence (AI) interaction scenarios and to enable personalized decision-making [188]. However, the heterogeneity of tabular data poses problems for counterfactual explanation methods to be reliably deployed in practice. The problem of efficiently handling heterogeneous tabular data in the presence of feasibility constraints remains unsolved [157].

6) Transfer of Deep Learning Methods to Data Streams: Recent work shows that some of the limitations of neural networks in an evolving data stream can be overcome [25], [209]. Conversely, changes in the parameters of a neural network may be effectively used to weigh the importance of input features over time [210] or to detect concept drift [211]. Accordingly, we argue that deep learning for streaming data—in particular strategies for dealing with evolving and heterogeneous tabular data—should receive more attention in the future.

7) Transfer Learning for Tabular Data: Reusing knowledge gained solving one problem and applying it to a different task is the research problem addressed by transfer learning. While transfer learning is successfully used in computer vision and natural language processing applications [212], there are no efficient and generally accepted ways to do transfer learning for tabular data. Hence, a general research question can be how to share knowledge between multiple (related) tabular datasets efficiently.

8) Data Augmentation for Tabular Data: Data augmentation has proven highly effective to prevent overfitting, especially in computer vision [213]. While some data augmentation techniques for tabular data exist, e.g., SMOTE-NC [214], simple models fail to capture the dependency structure of the data. Therefore, generating additional samples in a continuous latent space is a promising direction. This was investigated by Darabi and Elor [37] for minority oversampling. Nevertheless, the reported improvements are only marginal. Thus, future work is required to find simple, yet effective random transformations to enhance tabular training sets.

9) Self-Supervised Learning: Large-scale labeled data are usually required to train deep neural networks; however, data labeling is an expensive task. To avoid this expensive step, self-supervised methods propose to learn general feature representations from available unlabeled data. These methods have also shown astonishing results in computer vision and natural language processing [215], [216]. Only a few recent works in this direction [79], [80], [217] deal with heterogeneous data. Hence, novel self-supervised learning approaches dedicated to tabular data might be worth investigating.

IX. Conclusion

This survey is the first work to systematically explore deep neural network approaches for heterogeneous tabular data. In this context, we highlighted the main challenges and research advances in modeling, generating, and explaining tabular data. We introduced a unified taxonomy that categorizes deep learning approaches for tabular data into three branches: data transformation methods, specialized architectures, and regularization models. We believe that our taxonomy will help catalog future research and better understand and address the remaining challenges in applying deep learning to tabular data. We hope that it will help researchers and practitioners to find the most appropriate strategies and methods for their applications.

In addition, we also conducted an unbiased evaluation of the state-of-the-art deep learning approaches on multiple real-world datasets. Deep neural network-based methods for heterogeneous tabular data are still inferior to machine learning methods based on decision tree ensembles for small- and medium-sized datasets (less than $\sim$1M samples). Only for a very large dataset mainly consisting of continuous and numerical variables, the deep learning model SAINT outperformed these classical approaches. Furthermore, we assessed the explanation properties of deep learning models with the self-attention mechanism. Although the TabNet model shows promising explanatory capabilities, inconsistencies between the explanations remain an open issue.

Due to the importance of tabular data to industry and academia, new ideas in this area are in high demand and can have a significant impact. With this review, we hope to provide interested readers with the references and insights they need to address open challenges and effectively advance the field.

References

[1] J. Schmidhuber, “Deep learning in neural networks: An overview,” *Neural Netw.*, vol. 61, pp. 85–117, May 2015.

[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[3] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[4] K. Greff, R. K. Srivastava, J. Koutnìk, B. R. Steunebrink, and J. Schmidhuber, “LSTM: A search space Odyssey,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.

[5] A. Vaswani et al., “Attention is all you need,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[6] S. O. Arik and T. Pfister, “TabNet: Attentive interpretable tabular learning,” 2019, *arXiv:1908.07442*.

[7] S. Popov, S. Morozov, and A. Babenko, “Neural oblivious decision ensembles for deep learning on tabular data,” 2019, *arXiv:1909.06312*.

[8] R. Shwartz-Ziv and A. Armon, “Tabular data: Deep learning is not all you need,” 2021, *arXiv:2106.03253*.

[9] G. Somepalli, M. Goldblum, A. Schwarzschild, C. B. Bruss, and T. Goldstein, “SAINT: Improved neural networks for tabular data via row attention and contrastive pre-training,” 2021, *arXiv:2106.01342*.

[10] A. Kadra, M. Lindauer, F. Hutter, and J. Grabocka, “Well-tuned simple nets excel on tabular datasets,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 1–14.

[11] D. Ulmer, L. Meijerink, and G. Cinà, “Trust Issues: Uncertainty estimation does not enable reliable OOD detection on medical tabular data,” in *Proc. Mach. Learn. Health NeurIPS Workshop*, 2020, pp. 341–354.

[12] S. Somani et al., “Deep learning and the electrocardiogram: Review of the current state-of-the-art,” *EP Europace*, vol. 23, no. 8, pp. 1179–1191, Aug. 2021.

[13] V. Borisov, E. Kasneci, and G. Kasneci, “Robust cognitive load detection from wrist-band sensors,” *Comput. Hum. Behav. Rep.*, vol. 4, Aug. 2021, Art. no. 100116.

[14] J. M. Clements, D. Xu, N. Yousefi, and D. Efimov, “Sequential deep learning for credit risk monitoring with tabular financial data,” 2020, *arXiv:2012.15330*.

[15] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, “DeepFM: A factorization-machine based neural network for CTR prediction,” 2017, *arXiv:1703.04247*.

[16] Z. Shuai, L. Yao, A. Sun, and T. Yi, “Deep learning based recommender system: A survey and new perspectives,” *ACM Comput. Surv.*, vol. 52, no. 1, pp. 1–38, 2017.

[17] Q. Zhang, L. Cao, C. Shi, and Z. Niu, “Neural time-aware sequential recommendation by jointly modeling preference dynamics and explicit feature couplings,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 10, pp. 5125–5137, Oct. 2022.

[18] M. Ahmed, H. Afzal, A. Majeed, and B. Khan, “A survey of evolution in predictive models and impacting factors in customer churn,” *Adv. Data Sci. Adapt. Anal.*, vol. 9, no. 3, Jul. 2017, Art. no. 1750007.

[19] A. L. Buczak and E. Guven, “A survey of data mining and machine learning methods for cyber security intrusion detection,” *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1153–1176, 2nd Quart., 2016.

[20] F. Cartella, O. Anunciação, Y. Funabiki, D. Yamaguchi, T. Akishita, and O. Elshocht, “Adversarial attacks for tabular data: Application to fraud detection and imbalanced data,” in *Proc. CEUR Workshop*, vol. 2808, 2021, pp. 1–9.

[21] C. J. Urban and K. M. Gates, “Deep learning: A primer for psychologists,” *Psychol. Methods*, vol. 26, no. 6, pp. 743–773, 2021.

[22] G. Pang, C. Aggarwal, C. Shen, and N. Sebe, “Editorial deep learning for anomaly detection,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 6, pp. 2282–2286, Jun. 2022.

[23] S. Wang et al., "Multiview deep anomaly detection: A systematic exploration," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jun. 26, 2022, doi: 10.1109/TNNLS.2022.3184723.

[24] V. Škvára, J. Francå, M. Zorek, T. Pevnỳ, and V. Šmídl, "Comparison of anomaly detectors: Context matters," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 6, pp. 2494–2507, Jun. 2022.

[25] D. Sahoo, Q. Pham, J. Lu, and S. C. H. Hoi, "Online deep learning: Learning deep neural networks on the fly," 2017, *arXiv:1711.03705*.

[26] X. He, K. Zhao, and X. Chu, "AutoML: A survey of the state-of-the-art," *Knowl.-Based Syst.*, vol. 212, Jan. 2021, Art. no. 106622.

[27] P. Yin, G. Neubig, W.-T. Yih, and S. Riedel, "TaBERT: Pretraining for joint understanding of textual and tabular data," 2020, *arXiv:2005.08314*.

[28] Z. Wang, Q. She, and T. E. Ward, "Generative adversarial networks in computer vision: A survey and taxonomy," 2019, *arXiv:1906.01529*.

[29] D. Lichtenwalter, P. Burggräf, J. Wagner, and T. Weißer, "Deep multimodal learning for manufacturing problem solving," *Proc. CIRP*, vol. 99, pp. 615–620, 2021.

[30] T. Baltrušaitis, C. Ahuja, and L.-P. Morency, "Multimodal machine learning: A survey and taxonomy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 2, pp. 423–443, Feb. 2019.

[31] D. Medvedev and A. D'yakonov, "New properties of the data distillation method when working with tabular data," 2020, *arXiv:2010.09839*.

[32] J. Li, Y. Li, X. Xiang, S.-T. Xia, S. Dong, and Y. Cai, "TNT: An interpretable tree-network-tree learning framework using knowledge distillation," *Entropy*, vol. 22, no. 11, p. 1203, Oct. 2020.

[33] D. Roschewitz, M.-A. Hartley, L. Corinzia, and M. Jaggi, "IFedAvg: Interpretable data-interoperability for federated learning," 2021, *arXiv:2107.06580*.

[34] A. Sánchez-Morales, J.-L. Sancho-Gómez, J.-A. Martínez-García, and A. R. Figueiras-Vidal, "Improving deep learning performance with missing values via deletion and compensation," *Neural Comput. Appl.*, vol. 32, no. 17, pp. 13233–13244, Sep. 2020.

[35] M. Abroshan, K. H. Yip, C. Tekin, and M. Van Der Schaar, "Conservative policy construction using variational autoencoders for logged data with missing values," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jan. 10, 2022, doi: 10.1109/TNNLS.2021.3136385.

[36] J. Engelmann and S. Lessmann, "Conditional Wasserstein GAN-based oversampling of tabular data for imbalanced learning," *Expert Syst. Appl.*, vol. 174, Jul. 2021, Art. no. 114582.

[37] S. Darabi and Y. Elor, "Synthesising multi-modal minority samples for tabular data," 2021, *arXiv:2105.08204*.

[38] S. Kamthe, S. Assefa, and M. Deisenroth, "Copula flows for synthetic data generation," 2021, *arXiv:2101.00598*.

[39] E. Choi, S. Biswal, B. Malin, J. Duke, W. F. Stewart, and J. Sun, "Generating multi-label discrete patient records using generative adversarial networks," in *Proc. 2nd Mach. Learn. Healthcare Conf.*, 2017, pp. 286–305.

[40] State of California, Department of Justice. (2018). *California Consumer Privacy Act (CCPA)*. Accessed: Dec. 20, 2022. [Online]. Available: https://oag.ca.gov/privacy/ccpa

[41] GDPR. (2016). *Regulation (EU) 2016/679 of the European Parliament and of the Council*. Official Journal of the European Union. [Online]. Available: http://www.privacyregulation.eu/en/13.htm

[42] P. Voigt and A. Von Dem Bussche, "The EU general data protection regulation (GDPR)," in *A Practical Guide*, vol. 10, 1st ed. Cham, Switzerland: Springer, 2017, Art. no. 3152676.

[43] M. Sahakyan, Z. Aung, and T. Rahwan, "Explainable artificial intelligence for tabular data: A survey," *IEEE Access*, vol. 9, pp. 135392–135422, 2021.

[44] B. I. Grisci, M. J. Krause, and M. Dorn, "Relevance aggregation for neural networks interpretability and knowledge discovery on tabular data," *Inf. Sci.*, vol. 559, pp. 111–129, Jun. 2021.

[45] U. Bhatt et al., "Explainable machine learning in deployment," in *Proc. Conf. Fairness, Accountability, Transparency*, Jan. 2020, pp. 648–657.

[46] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 785–794.

[47] J. T. Hancock and T. M. Khoshgoftaar, "Survey on categorical data for neural networks," *J. Big Data*, vol. 7, no. 1, pp. 1–41, Dec. 2020.

[48] Y. Gorishniy, I. Rubachev, V. Khrulkov, and A. Babenko, "Revisiting deep learning models for tabular data," 2021, *arXiv:2106.11959*.

[49] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[50] L. Katzir, G. Elidan, and R. El-Yaniv, "Net-DNF: Effective deep modeling of tabular data," in *Proc. Int. Conf. Learn. Represent.*, 2021, pp. 1–16.

[51] R. U. David and M. Lane, *Introduction to Statistics*. 2003. [Online]. Available: http://onlinestatbook.com/

[52] M. Ryan, *Deep Learning With Structured Data*. New York, NY, USA: Simon & Schuster, 2020.

[53] M. W. Cvitkovic et al., "Deep learning in unconventional domains," Ph.D. dissertation, California Inst. Technol., Pasadena, CA, USA, 2020.

[54] D. Dua and C. Graff. (2017). *UCI Machine Learning Repository*. [Online]. Available: http://archive.ics.uci.edu/ml

[55] A. J. Miles, "The sunstroke epidemic of Cincinnati, Ohio, during the summer of 1881," *Public Health Papers Rep.*, vol. 7, no. 1, pp. 293–304, 1881.

[56] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Ann. Eugenics*, vol. 7, no. 2, pp. 179–188, Aug. 1936.

[57] D. A. Jdanov, D. Jasilionis, V. M. Shkolnikov, and M. Barbieri, "Human mortality database," in *Encyclopedia Gerontology Population Aging*, D. Gu and M. E. Dupre, Eds. Cham, Switzerland: Springer, 2020.

[58] E. Fix, *Discriminatory Analysis: Nonparametric Discrimination, Consistency Properties*. Wright-Patterson AFB, OH, USA: USAF school of Aviation Medicine, 1951.

[59] C. L. Giles, C. B. Miller, D. Chen, H. H. Chen, G. Z. Sun, and Y. C. Lee, "Learning and extracting finite state automata with second-order recurrent neural networks," *Neural Comput.*, vol. 4, no. 3, pp. 393–405, May 1992.

[60] L. Willenborg and T. De Waal, *Statistical Disclosure Control in Practice*, vol. 111. New York, NY, USA: Springer, 1996.

[61] M. Richardson, E. Dominowska, and R. Ragno, "Predicting clicks: Estimating the click-through rate for new ads," in *Proc. 16th Int. Conf. World Wide Web (WWW)*, 2007, pp. 521–530.

[62] G. Ke, Z. Xu, J. Zhang, J. Bian, and T.-Y. Liu, "DeepGBM: A deep learning framework distilled by GBDT for online prediction tasks," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 384–394.

[63] I. Shavitt and E. Segal, "Regularization learning networks: Deep learning for tabular datasets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 1379–1389.

[64] T. B. Brown et al., "Language models are few-shot learners," 2020, *arXiv:2005.14165*.

[65] A. Dosovitskiy et al., "An image is worth 16×16 words: Transformers for image recognition at scale," in *Proc. Int. Conf. Learn. Represent.*, 2021, pp. 1–11.

[66] S. Khan, M. Naseer, M. Hayat, S. Waqas Zamir, F. Shahbaz Khan, and M. Shah, "Transformers in vision: A survey," 2021, *arXiv:2101.01169*.

[67] G. Pang, C. Shen, L. Cao, and A. V. D. Hengel, "Deep learning for anomaly detection: A review," *ACM Comput. Surv.*, vol. 54, no. 2, pp. 1–38, Mar. 2021.

[68] A. F. Karr, A. P. Sanil, and D. L. Banks, "Data quality: A statistical perspective," *Stat. Methodol.*, vol. 3, no. 2, pp. 137–173, 2006.

[69] L. Xu and K. Veeramachaneni, "Synthesizing tabular data using generative adversarial networks," 2018, *arXiv:1811.11264*.

[70] G. Ke et al., "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 3146–3154.

[71] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "CatBoost: Unbiased boosting with categorical features," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 6638–6648.

[72] Y. Zhu et al., "Converting tabular data into images for deep learning with convolutional neural networks," *Sci. Rep.*, vol. 11, no. 1, pp. 1–11, May 2021.

[73] N. Rahaman et al., "On the spectral bias of neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 5301–5310.

[74] B. R. Mitchell et al., "The spatial inductive bias of deep learning," Ph.D. dissertation, Johns Hopkins Univ., Baltimore, MD, USA, 2017.

[75] Y. Gorishniy, I. Rubachev, and A. Babenko, "On embeddings for numerical features in tabular deep learning," 2022, *arXiv:2203.05556*.

[76] E. Fitkov-Norris, S. Vahid, and C. Hand, "Evaluating the impact of categorical data encoding and scaling on neural network classification performance: The case of repeat consumption of identical cultural goods," in *Proc. Int. Conf. Eng. Appl. Neural Netw.* Cham, Switzerland: Springer, 2012, pp. 343–352.

[77] D. Baylor et al., "TFX: A TensorFlow-based production-scale machine learning platform," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2017, pp. 1387–1395.

[78] B. Sun et al., "SuperTML: Two-dimensional word embedding for the precognition on structured tabular data," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2019, pp. 1–9.

[79] J. Yoon, Y. Zhang, J. Jordon, and M. van der Schaar, "VIME: Extending the success of self- and semi-supervised learning to tabular domaindim," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 1–11.

[80] D. Bahri, H. Jiang, Y. Tay, and D. Metzler, "SCARF: Self-supervised contrastive learning using random feature corruption," 2021, *arXiv:2106.15147*.

[81] H.-T. Cheng et al., "Wide & deep learning for recommender systems," in *Proc. 1st Workshop Deep Learn. Recommender Syst.*, 2016, pp. 7–10.

[82] N. Frosst and G. Hinton, "Distilling a neural network into a soft decision tree," 2017, *arXiv:1711.09784*.

[83] J. Lian, X. Zhou, F. Zhang, Z. Chen, X. Xie, and G. Sun, "XDeepFM: Combining explicit and implicit feature interactions for recommender systems," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1754–1763.

[84] G. Ke, J. Zhang, Z. Xu, J. Bian, and T.-Y. Liu. (2018). *TabNN: A Universal Neural Network Solution for Tabular Data*. [Online]. Available: https://openreview.net/forum?id=r1eJssCqY7

[85] R. Agarwal et al., "Neural additive models: Interpretable machine learning with neural nets," 2020, *arXiv:2004.13912*.

[86] Y. Luo, H. Zhou, W.-W. Tu, Y. Chen, W. Dai, and Q. Yang, "Network on network for tabular data classification in real-world applications," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2020, pp. 2317–2326.

[87] Z. Liu, Q. Liu, H. Zhang, and Y. Chen, "DNN2LR: Interpretation-inspired feature crossing for real-world tabular data," 2020, *arXiv:2008.09775*.

[88] S. Ivanov and L. Prokhorenkova, "Boost then Convolve: Gradient boosting meets graph neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2021, pp. 1–16.

[89] H. Luo, F. Cheng, H. Yu, and Y. Yi, "SDTR: Soft decision tree regressor for tabular data," *IEEE Access*, vol. 9, pp. 55999–56011, 2021.

[90] X. Huang, A. Khetan, M. Cvitkovic, and Z. Karnin, "TabTransformer: Tabular data modeling using contextual embeddings," 2020, *arXiv:2012.06678*.

[91] S. Cai, K. Zheng, G. Chen, H. V. Jagadish, B. C. Ooi, and M. Zhang, "ARM-Net: Adaptive relation modeling network for structured data," in *Proc. Int. Conf. Manage. Data*, Jun. 2021, pp. 207–220.

[92] J. Kossen, N. Band, C. Lyle, A. Gomez, T. Rainforth, and Y. Gal, "Self-attention between datapoints: Going beyond individual input-output pairs in deep learning," in *Proc. Adv. Neural Inf. Process. Syst.*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021, pp. 28742–28756.

[93] Y. Yamada, O. Lindenbaum, S. Negahban, and Y. Kluger, "Feature selection using stochastic gates," in *Proc. Mach. Learn. Syst.*, 2020, pp. 8952–8963.

[94] D. Micci-Barreca, "A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems," *ACM SIGKDD Explor. Newslett.*, vol. 3, no. 1, pp. 27–32, Jul. 2001.

[95] J. H. Friedman, "Stochastic gradient boosting," *Comput. Statist. Data Anal.*, vol. 38, no. 4, pp. 367–378, 2002.

[96] P. Langley and S. Sage, "Oblivious decision trees and abstract cases," in *Proc. Work. Notes AAAI Workshop Case-Based Reasoning*. Seattle, WA, USA, 1994, pp. 113–117.

[97] B. Peters, V. Niculae, and A. F. T. Martins, "Sparse Sequence-to-Sequence models," 2019, *arXiv:1905.05702*.

[98] Y. LeCun and C. Cortes. (2010). *MNIST Handwritten Digit Database*. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[99] C. Guo and F. Berkhahn, "Entity embeddings of categorical variables," 2016, *arXiv:1604.06737*.

[100] S. Rendle, "Factorization machines," in *Proc. IEEE Int. Conf. Data Mining*, Dec. 2010, pp. 995–1000.

[101] F. Moosmann, B. Triggs, and F. Jurie, "Fast discriminative visual codebooks using randomized clustering forests," in *Proc. 20th Annu. Conf. Neural Inf. Process. Syst. (NIPS)*. Cambridge, MA, USA: MIT Press, 2006, pp. 985–992.

[102] X. He et al., "Practical lessons from predicting clicks on ads at Facebook," in *Proc. 8th Int. Workshop Data Mining Online Advertising*, 2014, pp. 1–9.

[103] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Mar. 2020.

[104] C. Wang, M. Li, and A. J. Smola, "Language models with transformers," 2019, *arXiv:1904.09408*.

[105] A. F. T. Martins and R. Fernandez Astudillo, "From softmax to sparsemax: A sparse model of attention and multi-label classification," 2016, *arXiv:1602.02068*.

[106] G. Van Rossum and F. L. Drake, Jr., *Python Reference Manual*. Amsterdam, The Netherlands: Centrum voor Wiskunde en Informatica, 1995.

[107] M. Joseph, "PyTorch tabular: A framework for deep learning with tabular data," 2021, *arXiv:2104.13638*.

[108] S. Boughorbel, F. Jarray, and A. Kadri, "Fairness in TabNet model by disentangled representation for the prediction of hospital no-show," 2021, *arXiv:2103.04048*.

[109] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, "A survey on bias and fairness in machine learning," *ACM Comput. Surv.*, vol. 54, no. 6, pp. 1–35, Jul. 2021.

[110] S. Yun, D. Han, S. Chun, S. J. Oh, Y. Yoo, and J. Choe, "CutMix: Regularization strategy to train strong classifiers with localizable features," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6023–6032.

[111] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization," 2017, *arXiv:1710.09412*.

[112] V. Borisov, J. Haug, and G. Kasneci, "CancelOut: A layer for feature selection in deep neural networks," in *Proc. Int. Conf. Artif. Neural Netw.* Cham, Switzerland: Springer, 2019, pp. 72–83.

[113] G. Valdes, W. Arbelo, Y. Interian, and J. H. Friedman, "Lockout: Sparse regularization of neural networks," 2021, *arXiv:2107.07160*.

[114] J. Fiedler, "Simple modifications to improve tabular neural networks," 2021, *arXiv:2108.03214*.

[115] K. Lounici, K. Meziani, and B. Riu, "Muddling label regularization: Deep learning for tabular datasets," 2021, *arXiv:2106.04462*.

[116] S. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proc. NeurIPS*, 2017, pp. 1–10.

[117] H. Chen, S. Jajodia, J. Liu, N. Park, V. Sokolov, and V. S. Subrahmanian, "FakeTables: Using GANs to generate functional dependency preserving tables with bounded real data," in *Proc. Twenty-Eighth Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 2074–2080.

[118] L. Gondara and K. Wang, "MIDA: Multiple imputation using denoising autoencoders," in *Proc. Pacific–Asia Conf. Knowl. Discovery Data Mining*. Cham, Switzerland: Springer, 2018, pp. 260–272.

[119] R. D. Camino et al., "Working with deep generative models and tabular data imputation," in *Proc. ICML Artemiss Workshop*, 2020, pp. 1–6.

[120] M. Quintana and C. Miller, "Towards class-balancing human comfort datasets with GANs," in *Proc. 6th ACM Int. Conf. Syst. Energy-Efficient Buildings, Cities, Transp.*, Nov. 2019, pp. 391–392.

[121] A. Koivu, M. Sairanen, A. Airola, and T. Pahikkala, "Synthetic minority oversampling of vital statistics data with generative adversarial networks," *J. Amer. Med. Inform. Assoc.*, vol. 27, no. 11, pp. 1667–1674, Nov. 2020.

[122] J. Fan, J. Chen, T. Liu, Y. Shen, G. Li, and X. Du, "Relational data synthesis using generative adversarial networks: A design space exploration," *Proc. VLDB Endowment*, vol. 13, no. 12, pp. 1962–1975, Aug. 2020.

[123] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of StyleGAN," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 8110–8119.

[124] K. Lin, D. Li, X. He, Z. Zhang, and M.-T. Sun, "Adversarial ranking for language generation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1–11.

[125] S. Subramanian, S. Rajeswar, F. Dutil, C. Pal, and A. Courville, "Adversarial generation of natural language," in *Proc. 2nd Workshop Represent. Learn. NLP*, 2017, pp. 241–251.

[126] N. Patki, R. Wedge, and K. Veeramachaneni, "The synthetic data vault," in *Proc. IEEE Int. Conf. Data Sci. Adv. Analytics (DSAA)*, Oct. 2016, pp. 399–410.

[127] Z. Li, Y. Zhao, and J. Fu, "SynC: A copula based framework for generating synthetic data from aggregated sources," in *Proc. Int. Conf. Data Mining Workshops (ICDMW)*, Nov. 2020, pp. 571–578.

[128] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao, "PrivBayes: Private data release via Bayesian networks," *ACM Trans. Database Syst.*, vol. 42, no. 4, pp. 1–41, Oct. 2017.

[129] C. Chow and C. Liu, “Approximating discrete probability distributions with dependence trees,” *IEEE Trans. Inf. Theory*, vol. IT-14, no. 3, pp. 462–467, May 1968.

[130] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, “Modeling tabular data using conditional GAN,” in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2019, pp. 1–11.

[131] L. V. H. Vardhan and S. Kok, “Generating privacy-preserving synthetic tabular data using oblivious variational autoencoders,” in *Proc. Workshop Econ. Privacy Data Labor 37th Int. Conf. Mach. Learn.*, 2020, pp. 1–8.

[132] M. Baak, S. Brugman, I. F. Rojas, L. Dalmeida, R. E. Urlus, and J.-B. Oger, “Synthsonic: Fast, probabilistic modeling and synthesis of tabular data,” in *Proc. Int. Conf. Artif. Intell. Statist.*, 2022, pp. 4747–4763.

[133] I. J. Goodfellow et al., “Generative adversarial networks,” 2014, *arXiv:1406.2661*.

[134] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” in *Proc. 4th Int. Conf. Learn. Represent. (ICLR)*, 2016, pp. 1–16.

[135] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein generative adversarial networks,” in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 214–223.

[136] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, “Improved training of Wasserstein GANs,” in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 5769–5779.

[137] M. G. Bellemare et al., “The Cramér distance as a solution to biased Wasserstein gradients,” 2017, *arXiv:1705.10743*.

[138] R. D. Hjelm, A. P. Jacob, T. Che, A. Trischler, K. Cho, and Y. Bengio, “Boundary-seeking generative adversarial networks,” in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–17.

[139] A. Srivastava, L. Valkov, C. Russell, M. U. Gutmann, and C. Sutton, “VEEGAN: Reducing mode collapse in GANs using implicit variational learning,” in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 3310–3320.

[140] D. P. Kingma and M. Welling, “Auto-encoding variational Bayes,” in *Proc. 2nd Int. Conf. Learn. Represent. (ICLR), Conf. Track*, 2014, pp. 1–14.

[141] C. Ma, S. Tschiatschek, R. Turner, J. M. Hernández-Lobato, and C. Zhang, “VAEM: A deep generative model for heterogeneous mixed type data,” in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 1–11.

[142] N. Park, M. Mohammadi, K. Gorde, S. Jajodia, H. Park, and Y. Kim, “Data synthesis based on generative adversarial networks,” *Proc. VLDB Endowment*, vol. 11, no. 10, pp. 1071–1083, Jun. 2018.

[143] E. Jang, S. Gu, and B. Poole, “Categorical reparameterization with Gumbel-Softmax,” in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–13.

[144] B. Wen, L. O. Colon, K. P. Subbalakshmi, and R. Chandramouli, “Causal-TGAN: Generating tabular data using causal generative adversarial networks,” 2021, *arXiv:2104.10680*.

[145] J. Jordon, J. Yoon, and M. Van Der Schaar, “PATE-GAN: Generating synthetic data with differential privacy guarantees,” in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–21.

[146] N. M. Jebreel, J. Domingo-Ferrer, A. Blanco-Justicia, and D. Sánchez, “Enhanced security and privacy via fragmented federated learning,” *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Oct. 19, 2022, doi: 10.1109/TNNLS.2022.3212627.

[147] A. Mottini, A. Lheritier, and R. Acuna-Agost, “Airline passenger name record generation using generative adversarial networks,” 2018, *arXiv:1807.06657*.

[148] R. Camino, C. Hammerschmidt, and R. State, “Generating multi-categorical samples with generative adversarial networks,” in *Proc. ICML Workshop Theor. Found. Appl. Deep Generative Models*, 2018, pp. 1–7.

[149] M. K. Baowaly, C.-C. Lin, C.-L. Liu, and K.-T. Chen, “Synthesizing electronic health records using improved generative adversarial networks,” *J. Amer. Med. Inform. Assoc.*, vol. 26, no. 3, pp. 228–241, Mar. 2019.

[150] Z. Zhao, A. Kunar, H. Van der Scheer, R. Birke, and L. Y. Chen, “CTAB-GAN: Effective table data synthesizing,” 2021, *arXiv:2102.08369*.

[151] V. Borisov, K. Seßler, T. Leemann, M. Pawelczyk, and G. Kasneci, “Language models are realistic tabular data generators,” 2022, *arXiv:2210.06280*.

[152] F. J. Massey, Jr., “The Kolmogorov-Smirnov test for goodness of fit,” *J. Amer. Statist. Assoc.*, vol. 46, no. 253, pp. 68–78, 1951.

[153] E. Tjoa and C. Guan, “A survey on explainable artificial intelligence (XAI): Toward medical XAI,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 11, pp. 4793–4813, Nov. 2021.

[154] J. Kauffmann, M. Esders, L. Ruff, G. Montavon, W. Samek, and K.-R. Müller, “From clustering to cluster explanations via neural networks,” *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jul. 7, 2022, doi: 10.1109/TNNLS.2022.3185901.

[155] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi, “A survey of methods for explaining black box models,” *ACM Comput. Surv.*, vol. 51, no. 5, pp. 1–42, Sep. 2019.

[156] K. Gade, S. C. Geyik, K. Kenthapadi, V. Mithal, and A. Taly, “Explainable AI in industry,” in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 3203–3204.

[157] M. Pawelczyk, S. Bielawski, J. Van Den Heuvel, T. Richter, and G. Kasneci, “CARLA: A Python library to benchmark algorithmic recourse and counterfactual explanation algorithms,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS) Benchmark Datasets Track*, 2021, pp. 1–22.

[158] Y. Lou, R. Caruana, and J. Gehrke, “Intelligible models for classification and regression,” in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2012, pp. 150–158.

[159] D. Alvarez-Melis and T. S. Jaakkola, “Towards robust interpretability with self-explaining neural networks,” in *Proc. NeurIPS*, 2018, pp. 1–10.

[160] D. Wang, Q. Yang, A. Abdul, and B. Y. Lim, “Designing theory-driven user-centric explainable AI,” in *Proc. CHI*, 2019, pp. 1–15.

[161] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, “On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation,” *PLoS ONE*, vol. 10, no. 7, Jul. 2015, Art. no. e0130140.

[162] G. Montavon, A. Binder, S. Lapuschkin, W. Samek, and K.-R. Müller, “Layer-wise relevance propagation: An overview,” in *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Cham, Switzerland: Springer, 2019, pp. 193–209.

[163] G. Kasneci and T. Gottron, “LICON: A linear weighting scheme for the contribution ofInput variables in deep artificial neural networks,” in *Proc. 25th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2016, pp. 45–54.

[164] M. Sundararajan, A. Taly, and Q. Yan, “Axiomatic attribution for deep networks,” in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 3319–3328.

[165] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, “Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks,” in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2018, pp. 1–9.

[166] M. T. Ribeiro, S. Singh, and C. Guestrin, “‘Why should i trust you?’: Explaining the predictions of any classifier,” in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 1135–1144.

[167] M. T. Ribeiro, S. Singh, and C. Guestrin, “Anchors: High-precision model-agnostic explanations,” in *Proc. AAAI*, 2018, pp. 1–9.

[168] S. M. Lundberg et al., “From local explanations to global understanding with explainable AI for trees,” *Nature Mach. Intell.*, vol. 2, pp. 56–67, Jan. 2020.

[169] J. Haug, S. Zürn, P. El-Jiz, and G. Kasneci, “On baselines for local feature attributions,” 2021, *arXiv:2101.00905*.

[170] Y. Liu, S. Khandagale, C. White, and W. Neiswanger, “Synthetic benchmarks for scientific research in explainable machine learning,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS) Benchmark Datasets Track*, 2021, pp. 1–25.

[171] S. Wachter, B. Mittelstadt, and C. Russell, “Counterfactual explanations without opening the black box: Automated decisions and the GDPR,” *Harvard J. Law Technol.*, vol. 31, no. 2, p. 841, 2018.

[172] B. Ustun, A. Spangher, and Y. Liu, “Actionable recourse in linear classification,” in *Proc. Conf. Fairness, Accountability, Transparency*, Jan. 2019, pp. 10–19.

[173] C. Russell, “Efficient search for diverse coherent explanations,” in *Proc. Conf. Fairness, Accountability, Transparency*, Jan. 2019, pp. 20–28.

[174] K. Rawal and H. Lakkaraju, “Beyond individualized recourse: Interpretable and interactive summaries of actionable recourses,” in *Proc. NeurIPS*, 2020, pp. 12187–12198.

[175] A.-H. Karimi, G. Barthe, B. Balle, and I. Valera, “Model-agnostic counterfactual explanations for consequential decisions,” in *Proc. Int. Conf. Artif. Intell. Statist.*, 2020, pp. 895–905.

[176] A. Dhurandhar et al., “Explanations based on the missing: Towards contrastive explanations with pertinent negatives,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2018, pp. 1–12.

[177] B. Mittelstadt, C. Russell, and S. Wachter, "Explaining explanations in AI," in *Proc. Conf. Fairness, Accountability, Transparency*, Jan. 2019, pp. 279–288.

[178] M. Pawelczyk, K. Broelemann, and G. Kasneci, "Learning model-agnostic counterfactual explanations for tabular data," in *Proc. Web Conf.*, Apr. 2020, pp. 3126–3132.

[179] M. Downs, J. L. Chu, Y. Yacoby, F. Doshi-Velez, and W. Pan, "CRUDS: Counterfactual recourse using disentangled subspaces," in *Proc. ICML Workshop Hum. Interpretability Mach. Learn. (WHI)*, 2020, 1–23.

[180] S. Joshi, O. Koyejo, W. Vijitbenjaronk, B. Kim, and J. Ghosh, "Towards realistic individual recourse and actionable explanations in black-box decision making systems," 2019, *arXiv:1907.09615*.

[181] D. Mahajan, C. Tan, and A. Sharma, "Preserving causal constraints in counterfactual explanations for machine learning classifiers," 2019, *arXiv:1912.03277*.

[182] M. Pawelczyk, K. Broelemann, and G. Kasneci, "On counterfactual explanations under predictive multiplicity," in *Proc. Conf. Uncertainty Artif. Intell. (UAI)*, 2020, pp. 809–818.

[183] A. Nazábal, P. M. Olmos, Z. Ghahramani, and I. Valera, "Handling incomplete heterogeneous data using VAEs," *Pattern Recognit.*, vol. 107, Nov. 2020, Art. no. 107501.

[184] S. Upadhyay, S. Joshi, and H. Lakkaraju, "Towards robust and reliable algorithmic recourse," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 34, 2021, pp.16926–16937.

[185] R. Dominguez-Olmedo, A.-H. Karimi, and B. Schölkopf, "On the adversarial robustness of causal algorithmic recourse," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2022, pp. 5324–5342.

[186] M. Pawelczyk, T. Datta, J. Van-Den-Heuvel, G. Kasneci, and H. Lakkaraju, "Probabilistically robust recourse: Navigating the trade-offs between costs and robustness in algorithmic recourse," 2022, *arXiv:2203.06768*.

[187] A.-H. Karimi, G. Barthe, B. Schölkopf, and I. Valera, "A survey of algorithmic recourse: Definitions, formulations, solutions, and prospects," 2020, *arXiv:2010.04050*.

[188] S. Verma, J. Dickerson, and K. Hines, "Counterfactual explanations for machine learning: A review," 2020, *arXiv:2010.10596*.

[189] A. Krizhevsky, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2009. [Online]. Available: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf

[190] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[191] FICO. (2019). *Home Equity Line of Credit (HELOC) Dataset*. Accessed: Jun. 15, 2022. [Online]. Available: https://community.fico.com/s/explainable-machine-learning-challenge

[192] P. Baldi, P. Sadowski, and D. Whiteson, "Searching for exotic particles in high-energy physics with deep learning," *Nature Commun.*, vol. 5, no. 1, pp. 1–9, Sep. 2014.

[193] C. Z. Mooney, *Monte Carlo Simulation*. Newbury Park, CA, USA: SAGE, 1997.

[194] R. K. Pace and R. Barry, "Sparse spatial autoregressions," *Statist. Probab. Lett.*, vol. 33, pp. 291–297, May 1997.

[195] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification Regression Trees*. Evanston, IL, USA: Routledge, 2017.

[196] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[197] K. Broelemann and G. Kasneci, "A gradient-based split criterion for highly accurate and transparent model trees," in *Proc. IJCAI*, 2019, pp. 1–8.

[198] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bull. Math. Biophys.*, vol. 5, no. 4, pp. 115–133, 1943.

[199] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 1–10.

[200] D. Merkel, "Docker: Lightweight Linux containers for consistent development and deployment," *Linux J.*, vol. 2014, no. 239, p. 2, 2014.

[201] C. S. Bojer and J. P. Meldgaard, "Kaggle forecasting competitions: An overlooked learning opportunity," *Int. J. Forecasting*, vol. 37, no. 2, pp. 587–603, Apr. 2021.

[202] Y. Rong, T. Leemann, V. Borisov, G. Kasneci, and E. Kasneci, "A consistent and efficient evaluation strategy for attribution methods," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 18770–18795.

[203] R. Tomsett, D. Harborne, S. Chakraborty, P. Gurram, and A. Preece, "Sanity checks for saliency metrics," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 4, 2020, pp. 6021–6029.

[204] E. Ntoutsi et al., "Bias in data-driven artificial intelligence systems-an introductory survey," *Wiley Interdiscipl. Reviews: Data Mining Knowl. Discovery*, vol. 10, no. 3, p. e1356, 2020.

[205] A. Giloni et al., "BENN: Bias estimation using a deep neural network," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, May 11, 2022, doi: 10.1109/TNNLS.2022.3172365.

[206] Y. Sun, K. Tang, Z. Zhu, and X. Yao, "Concept drift adaptation by exploiting historical knowledge," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 10, pp. 4822–4832, Oct. 2018.

[207] P. Domingos and G. Hulten, "Mining high-speed data streams," in *Proc. 6th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2000, pp. 71–80.

[208] C. Manapragada, G. I. Webb, and M. Salehi, "Extremely fast decision tree," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1953–1962.

[209] P. Duda, M. Jaworski, A. Cader, and L. Wang, "On training deep neural networks using a streaming approach," *J. Artif. Intell. Soft Comput. Res.*, vol. 10, no. 1, pp. 15–26, Jan. 2020.

[210] J. Haug, M. Pawelczyk, K. Broelemann, and G. Kasneci, "Leveraging model inherent variable importance for stable online feature selection," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 1478–1502.

[211] J. Haug and G. Kasneci, "Learning parameter distributions to detect concept drift in data streams," in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Jan. 2021, pp. 9452–9459.

[212] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," in *Proc. Int. Conf. Artif. Neural Netw.* Cham, Switzerland: Springer, 2018, pp. 270–279.

[213] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *J. Big Data*, vol. 6, no. 1, pp. 1–48, Dec. 2019.

[214] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, vol. 16, no. 1, pp. 321–357, Jan. 2002.

[215] L. Jing and Y. Tian, "Self-supervised visual feature learning with deep neural networks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 11, pp. 4037–4058, Nov. 2021.

[216] X. Liu et al., "Self-supervised learning: Generative or contrastive," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 1, pp. 857–876, Jan. 2021.

[217] T. Ucar, E. Hajiramezanali, and L. Edwards, "SubTab: Subsetting features of tabular data for self-supervised representation learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 1–13.

A.1.2 DeepTLF: Robust Deep Neural Networks for Heterogeneous Tabular Data

Publication: Published in the International Journal of Data Science and Analytics journal, 2022.

Contribution: I was the primary developer of the DeepTLF framework and conducted the experiments. Additionally, I also performed the experiments and wrote most parts of the paper. Klaus Broelemann, Gjergji Kasneci, and Enkelejda Kasneci provided valuable input and feedback throughout the process, helping to refine and improve the ideas, formalizations, and analysis in the paper. All co-authors played a role in reviewing and revising the final manuscript.

REGULAR PAPER

DeepTLF: robust deep neural networks for heterogeneous tabular data

Vadim Borisov[1] · Klaus Broelemann[2] · Enkelejda Kasneci[1] · Gjergji Kasneci[1,2]

Received: 8 February 2022 / Accepted: 16 July 2022

Abstract
Although deep neural networks (DNNs) constitute the state of the art in many tasks based on visual, audio, or text data, their performance on heterogeneous, tabular data is typically inferior to that of decision tree ensembles. To bridge the gap between the difficulty of DNNs to handle tabular data and leverage the flexibility of deep learning under input heterogeneity, we propose *DeepTLF*, a framework for deep tabular learning. The core idea of our method is to transform the heterogeneous input data into homogeneous data to boost the performance of DNNs considerably. For the transformation step, we develop a novel knowledge distillations approach, *TreeDrivenEncoder*, which exploits the structure of decision trees trained on the available heterogeneous data to map the original input vectors onto homogeneous vectors that a DNN can use to improve the predictive performance. Within the proposed framework, we also address the issue of the multimodal learning, since it is challenging to apply decision tree ensemble methods when other data modalities are present. Through extensive and challenging experiments on various real-world datasets, we demonstrate that the DeepTLF pipeline leads to higher predictive performance. On average, our framework shows 19.6% performance improvement in comparison to DNNs. The DeepTLF code is publicly available.

Keywords Deep neural networks · Heterogeneous data · Tabular data · Tabular data encoding · Multimodal learning

1 Introduction

Tabular data is the most commonly used form of data, and it is ubiquitous in various applications [1], such as medical diagnosis based on patient history [2], predictive analytics for financial applications [3], cybersecurity [4]. Although deep neural networks (DNNs) perform outstandingly well on homogeneous data, e.g., visual, audio, and textual data [5], heterogeneous, tabular data still pose a challenge to these models [1,6].

We hypothesize that the moderate performance of DNNs on tabular data comes from two major factors. The first is the inductive bias(es) [7,8]; for example, convolutional neural networks (CNNs) assume that specific spatial structures are present in the data, recurrent neural networks (RNNs) assume that a temporal relationship between data points exists, whereas tabular data do not have any spatial or temporal connections. The second reason is the high information loss during the data preprocessing step since tabular input data need to undergo cleansing (dealing with missing, noisy, and inconsistent values), uniform discretized representation (handling categorical and continuous values together), and scaling (standardized representation of features) steps. Along with these feature-processing steps, important information contained in the data may get lost, and hence, the preprocessed feature vectors[1](especially when one-hot encoded) may negatively impact training and learning effectiveness [9]. As reported in [10], an efficient transformation of categorical data for training DNNs is still a significant challenge. Furthermore, a work [11] shows that the embeddings (transformations) for numerical features can be also beneficial for DNNs.

Typically, when heterogeneous tabular data is involved, the first choice across all machine learning (ML) algorithms is ensemble models based on decision trees [12], such as random forests (RF) [13], or gradient-boosted decision trees (GBDT) [14]. Since the inductive bias(es) of the methods based on decision trees are well suited to non-spatial heterogeneous data, the data preprocessing step is reduced to a

Vadim Borisov
vadim.borisov@uni-tuebingen.de

[1] The University of Tübingen, Tübingen, Germany

[2] SCHUFA Holding AG, Wiesbaden, Germany

[1] In this work, we define a *feature vector* as an n-dimensional vector of numerical and categorical features that represent a data object.

Published online: 23 August 2022

Springer

minimum. In particular, the most common implementations of the GBDT algorithm—XGBoost [15], LightGBM [16], and CatBoost [17]—handle the missing values internally by searching for the best approximation of missing data points.

However, the most significant computational disadvantage of the decision tree-based methods is while training the need to store (almost) the entire dataset in memory [8]. Furthermore, in the multimodal datasets in which different data types are involved (e.g., visual and tabular data), decision tree-based models are not able to provide state-of-the-art results, whereas DNNs models allow for batch-learning (no need to store the whole dataset), and for those multimodal data tasks, DNNs demonstrate state-of-the-art performance [18].

Towards the goal of significantly boosting DNNs on tabular data, we propose DeepTLF, a novel deep tabular learning framework that exploits the advantages of the GBDT algorithm as well as the flexibility of DNNs. The key element of the framework is a novel encoding algorithm, TreeDrivenEncoder, which transforms the heterogeneous tabular data into homogeneous data by distilling knowledge from nodes of trained decision trees. Thus, DeepTLF can preserve most of the information that is contained in the original data and encoded in the structure of the decision trees and benefit from preprocessing power of decision tree-based algorithms.

Through experiments on various freely available real-world datasets, we demonstrate the advantages of such a composite learning approach for different prediction tasks. We argue that by transforming heterogeneous tabular data into homogeneous vectors, we can drastically improve the performance of DNNs on tabular data.

The main contributions of this work are: (I) We propose a deep tabular learning framework—DeepTLF—that combines the *preprocessing strengths* of GBDTs with the *learning flexibility* of DNNs. (II) The proposed framework builds on a generic approach for transforming heterogeneous tabular data into homogeneous vectors using the structure of decision trees from a gradient boosting model using a novel encoding function—TreeDrivenEncoder. Hence, the transformation approach can also be used independently from the presented deep learning framework. (III) In extensive experiments on eight datasets and compared with state-of-the-art ML approaches, we show that the proposed framework mitigates well-known data-processing challenges and leads to unprecedented predictive performance, outperforming all the competitors. (IV) For multimodal settings with tabular data, we demonstrate the robust performance of our deep tabular learning framework. (V) We provide an open-source implementation of proposed algorithm and published it online https://github.com/unnir/DeepTLF.

2 Related work

In recent years, deep neural networks on tabular data have received much attention from the machine learning and data science communities [1,8,19–32]. The existing approaches can be grouped into two broad categories— *architecture-based* and *data transformation-based* models.

Architecture-based models This group aims at developing new deep learning architectures for heterogeneous data [19,20,23,24,26]. For example, the authors of [23] proposed distinct neural network architecture for reducing the preprocessing and feature engineering effort by introducing a data sharing strategy between a deep and a wide network so that low- and high-level interactions between the inputs can be learned simultaneously, based on the ideas of factorization machines (FM) proposed in [33]. The work [34] extended the sharing strategy using the FM for structured data further. In [24], the authors propose an integrated solution by introducing two special neural networks, one for handling categorical features and another for numerical data. However, for mentioned approaches [23,24,34], it is not clear how other data-related issues, such as missing values, different scaling of numeric features, and noise, influence the predictions produced by the models.

Another line of research in this group tries to combine the advantages of decision trees and neural networks. For example, the authors of [35] introduced the neural decision forest algorithm, an ensemble of neural decision trees, where split functions in each tree node are randomized multilayer perceptrons (MLPs). Another approach [36] presented a strategy for selecting paths in a neural directed acyclic graph to produce the prediction for a given input. Hence, the selected neural paths are specialized to specific inputs. In [37], the authors empirically showed that neural networks with random forest structure could have better generalization ability across various input domains.

A fully differentiable architecture for deep learning, which generalizes ensembles of oblivious decision trees on tabular, is introduced in [20]. Their architecture (coined NODE) employs the entmax transformation [38] and thus maps a vector of real-valued scores to a discrete probability distribution. Furthermore, the work [8] promotes localized decisions that are taken over small subsets of the features.

Other approaches focus on architectures that build on attention-based (deep transformers) mechanisms [39]. For example, the authors of [19] and [22] propose an attentive transformer architecture for deep tabular learning. Their architecture also offers the possibility to interpret the input features; however, for reliable performance, a large amount

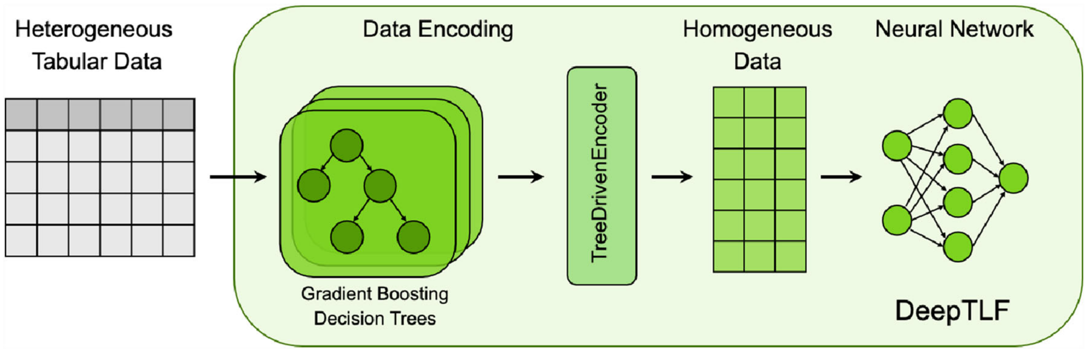


Fig. 1 A data pipeline for the DeepTLF framework. First, the *train* data is used to train a gradient-boosted decision trees (GBDT) model. The heterogeneous data is transformed by exploiting the structures of the decision trees in the ensemble. More specifically, the *TreeDrivenEncoder* algorithm distills information from trained decision trees of the GBDT model to produce homogeneous binary vectors. These vectors are then used to train a DNN. Note that DeepTLF does not require data preprocessing, such as normalization, handling missing values, and encoding categorical features; therefore, in total, it dramatically speeds up the *data preprocessing* time. Note that the test data are not used to train the GBDT algorithm

of training data is needed. Another drawback is that the attention mechanism is only applied to categorical data. Hence, the continuous data do not throw the self-attention block, meaning that correlations between categorical and continuous features are dropped. The work [27] proposes a variation of a transformer and offers semi-supervised learning. However, no clear statements can be drawn for all methods described so far regarding the relationship between data heterogeneity and prediction quality (especially robustness under noisy data or labels). Moreover, many of the solutions in this line of research are quite challenging from a practical perspective since it is often unclear which architectural choices should be employed in realistic scenarios.

These architecture-based approaches generally rely on novel neural network architectures, which are difficult to (re-)implement and optimize for specific real-world use cases. Especially for critical, data-intensive applications, e.g., data streaming, large-scale recommendation systems [40], and many more, it is not always clear what additional adjustments to the working pipeline are needed.

Data transformation-based models Another way to improve the predictive quality in the presence of tabular data is to transform heterogeneous data into homogeneous feature vectors. The transformation can range from simple data preprocessing, such as the normalization of numerical variables or binary encoding of categorical variables, to linear or nonlinear embedding schemes (e.g., generated by advanced autoencoders) [9,10]. The advantage of such data transformation approaches is that they do not require adapting the deep learning architecture. However, they may reduce the information content by smoothing critical values that might have been highly relevant for the final prediction.

Independent works [41,42] demonstrate that data can be encoded using the RF algorithm by accessing leaf indices in the decision trees. The idea was also utilized by [25], where trees from a GBDT model are used for the categorical data encoding instead of the RF model. These works show that the decision trees are a powerful and convenient way to implement nonlinear and categorical feature transformations for heterogeneous data. The DeepGBM framework [24] further evolved the idea of distilling knowledge from the decision tree leaf index by encoding them using a neural network for online learning tasks. Overall, the leaf embedding approach received much attention; however, *the leaf indices from a decision tree embedding do not fully represent the whole decision tree structure*. Thus, each boosted tree is treated as a new *meta categorical feature*, which might be an issue for the DNNs [10].

In contrast to related methods, our aim is to *holistically* distill the information from decision trees by utilizing the whole decision tree, not only the output leaves. The DeepTLF combines the advantages of GBDT (such as handling missing values and categorical variables) with the learning flexibility of DNNs to achieve superior and robust prediction performance. Also, [43] demonstrates that a DNN trained using distilled data can outperform models trained on the original data.

Other approaches such as NODE [20] and Net-DNF [8] try to mimic the decision trees using DNNs. Also, the work [44] proposes a gradient-descent-based strategy that exploits the decision tree structure to propagate gradients in the learning process. Our approach is different because DeepTLF is more robust to data inconsistencies and does not require new DNN architectures. Hence, it is straightforward to use.

Furthermore, the observation that local Boolean features from decision trees model can be informative for global modeling is also reported in [45], where the authors exploit sparse local contrastive explanations of a black-box model to obtain

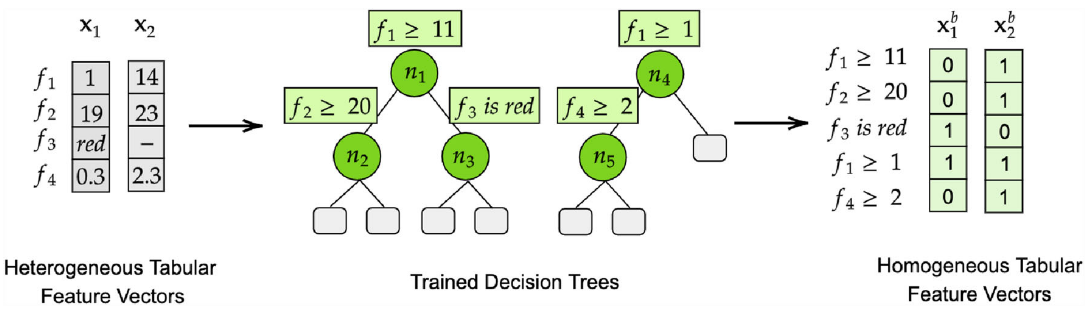


Fig. 2 A toy example of the proposed data transformation performed by the *TreeDrivenEncoder* algorithm. On the left, we see two heterogeneous input feature vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ from the original dataset $\mathcal{D}$, where $\mathbf{x}_i \in \mathbb{R} \times \mathbb{R} \times \{red, -\} \times \mathbb{R}$; f_1, f_2, f_4 are numerical features and f_3 is a categorical feature in a tabular dataset. Note there is also a missing value $-$ for the feature f_3 in $\mathbf{x}_2$. To encode the input data, we use two trained decision trees on the dataset $\mathcal{D}$, with 5 inner nodes in total. By evaluating the Boolean function in each inner node for a given input vector, we construct two homogeneous feature vectors $\mathbf{x}_1^b$ and $\mathbf{x}_2^b$, where a component of these vectors is set to 1 if the corresponding Boolean function evaluates to true and 0 otherwise (colour figure online)

custom Boolean features. A globally transparent model is then trained on the Boolean features; empirically, the global model shows a predictive performance that is slightly worse than state-of-the-art approaches.

In summary, in contrast to state-of-the-art methods that exploit decision tree structures and mainly focus on leaf indices, DeepTLF *utilizes the whole decision tree structure* from a GBDT model, and it furthermore *considers the representation of each feature independently* in the information distillation process. Our framework combines the advantages of gradient-boosted trees (such as handling different scales, different attribute types, missing values, outliers, and many more) with the learning flexibility of neural networks to achieve excellent predictive performance.

3 DeepTLF: deep tabular learning framework

In this section, we present the main components of our DeepTLF framework. As it is depicted in Fig. 1, DeepTLF consists of three major parts: (1) an ensemble of decision trees (in this work, we utilize the GBDT algorithm), (2) a TreeDrivenEncoder that performs the transformation of the original data into homogeneous, binary feature vectors by distilling the information contained in the structures of the decision trees through the TreeDrivenEncoder algorithm, and (3) a deep neural network model trained on the binary feature vectors obtained from the TreeDrivenEncoder algorithm. We will describe the details of each component in the following subsections.

3.1 Gradient-boosted decision tree

For the data encoding step, we selected one of the most powerful algorithms on tabular data, namely the gradient-boosted decision trees (GBDT) algorithm [14]. GBDT is a well-known and widely used ensemble algorithm for tabular data both in research and industrial applications [15] and is particularly successful for tasks containing heterogeneous features, small dataset sizes, and "noisy" data [12]. Especially when it comes to handling variance and bias, gradient boosting ensembles show highly competitive performance in comparison with state-of-the-art learning approaches [12,14]. In addition, multiple evaluations have empirically demonstrated that the *decision trees of a GBDT ensemble preserve the information* from the original data and can be used for further data processing [24,25].

The key idea of the GBDT algorithm is to construct a strong model by iterative addition of weak learners. The set of weak learners $\mathcal{H}$ is usually formed by shallow decision trees, which are directly trained on the original data. Consequently, almost no data preparation is needed, and the information loss is minimized. We denote a GBDT model as a set of decision trees:

$$\mathcal{T}_{\text{GBDT}} = \{T_1, T_2, \ldots, T_k\},$$

where k is the number of estimators in the GBDT algorithm.

The formal definition of the GBDT algorithm is in Appendix A.

3.2 Knowledge distillation from decision trees

The trained GBDT model provides structural data information, which also encodes dependencies between the input features with respect to the prediction task. In order to distill the knowledge from a tree-based model, we propose a novel data transformation algorithm – *TreeDrivenEncoder*. For every input vector from the original data, the proposed encoding method maps all features occurring in the decision trees of the GBDT ensemble to a binary feature vector $\mathbf{x}^b$.

This has the advantage that the neural network in the final component can form its own feature representations from *homogeneous data*. In Fig. 2, we illustrate the transformation obtained by applying the TreeDrivenEncoder algorithm on a toy example. There we have two input feature vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ with categorical and numerical values that are encoded into corresponding homogeneous binary feature vectors $\mathbf{x}_1^b$ and $\mathbf{x}_2^b$.

To formally describe the TreeDrivenEncoder algorithm, we first need a definition of the decision trees:

Definition 1 [Decision Tree] Let T be a structure $T = (V, E, \mu)$, where V is a set of nodes, $E \subseteq V \times V$ is a set of edges and $\mu = (\mu_v)_{v \in V}$ is a sequence of mapping functions $\mu_v : \mathbb{R}^{d \to V \cup \{\emptyset\}}$ that map input vectors to (child) nodes. We call T a (binary) decision tree if it satisfies the following properties:

1. (V, E) is a directed acyclic graph
2. There is exactly one designated node $v_r \in V$, called the root, which has no entering edges, i.e., for a node $v \in V$:

$$v = v_r \Leftrightarrow \forall w \in V : (w, v) \notin E.$$

3. Every node $v \in V \backslash \{v_r\}$ has *exactly one* entering edge with the parent node at its other end:

$$w \in V : (w, v) \in E \Leftrightarrow w = parent(v)\,.$$

4. Each node has either two or zero outgoing edges. We call the nodes with two outgoing edges inner nodes and all others nodes leaves. We denote the sets of inner nodes and leaves with V_I and V_L, respectively.
5. μ_v maps feature vectors from inner nodes to their child nodes and from leaves to $\emptyset$.

$$v \in V_I \Rightarrow \forall \mathbf{x} \in \mathbb{R}^d : (v, \mu_v(\mathbf{x})) \in E, \tag{1}$$

$$v \in V_L \Rightarrow \forall \mathbf{x} \in \mathbb{R}^d : \mu_v(\mathbf{x}) = \emptyset. \tag{2}$$

In the following, we denote the number of inner nodes as $|T| = V_I$. Furthermore, we assume that the child nodes can be identified as left or right child. For each inner node $v \in V_I$, we use a modified mapping function $\tilde{\mu}_v : \mathbb{R}^d \to \{0, 1\}$ (i.e., a Boolean function) where 0 encodes the left child and 1 encodes the right child.

For an input vector $\mathbf{x} \in \mathbb{R}^d$, we exploit the structure of T to derive a binary vector of length $|T|$. To this end, as shown in Alg. 1, we employ a breadth-first-search approach on the nodes of T. More specifically, for every feature that is evaluated at an inner node v of T, we retrieve the corresponding value from $\mathbf{x}$ and evaluate that value at v based on the associated Boolean function. Note that other node visiting strategies (e.g., depth-first search) can be used as well. It is only important that the order of $\tilde{\mu}_v$ is the same.

Finally, we concatenate all the vectors generated from the single decision trees of the ensemble $\mathcal{T}$ on the input vector $\mathbf{x}$, which gives us the final binary representation $\mathbf{x}^b$ of $\mathbf{x}$. We summarize the full algorithm in Alg. 1.

For mathematical completeness, the mapping obtained by applying TreeDrivenEncoder is formalized as follows. Given the feature vector $\mathbf{x}$ that represents an instance from the training dataset $\mathcal{D}$ and a trained decision tree ensemble $\mathcal{T}$ (i.e., a collection of decision trees) on the same dataset, we exploit the structure of each tree $T \in \mathcal{T}$ to produce a binary feature vector for the original feature vector $\mathbf{x} = (x_1, \ldots, x_d)^\top$ and employ a transformation function:

$$map_T : \mathbb{R}^d \to \{0, 1\}^{|T|}, \tag{3}$$

$$map_T : \mathbf{x} \mapsto (\tilde{\mu}_v(\mathbf{x}))_{v \in V_I}, \tag{4}$$

where V_I again represents the inner nodes in a well-defined order and $|T|$ their number. The mapping is performed such that at an inner node v of T, the corresponding component x_j of $\mathbf{x}$ *is mapped to 1 if the Boolean function at v evaluates to true for x_j and 0 otherwise*. Note that we apply the transformation function to each node in the decision tree T, even if a node does not belong to the *decision path* of $\mathbf{x}$; hence, it holds that $map_T(\mathbf{x}) \in \{0, 1\}^{|T|}$.

For the multiple decision trees T_1, ..., T_k, we construct a function:

$$\texttt{TreeDrivenEncoder} : \mathbb{R}^d \to \{0, 1\}^{\sum_{i=1}^{k} |T_i|}, \tag{5}$$

with

$$\texttt{TreeDrivenEncoder}(\mathbf{x}) = \big(map_{T_1}(\mathbf{x}), ..., map_{T_k}(\mathbf{x})\big)^\top. \tag{6}$$

3.3 Deep learning models for encoded homogeneous data

After the data distillation by the TreeDrivenEncoder algorithm, the new binary representations of the feature vectors are used to train and validate a chosen neural network.

A deep neural network defines a mapping function $\hat{f}$:

$$\hat{y} = f(x^b) \approx \hat{f}(x^b; W), \tag{7}$$

where $\hat{y}$ is the output of the deep tabular learning framework, x^b is a homogeneous tabular data transformed using TreeDrivenEncoder, and W are learning parameters of the deep learning model.

Depending on a downstream task, the deep learning architecture of the proposed framework should be selected. The

Algorithm 1 For a GBDT model $\mathcal{T}$ and an instance **x** from the underlying dataset, the *TreeDrivenEncoder* procedure visits the inner nodes of each $T \in \mathcal{T}$ (in a breadth-first search manner) and exploits their Boolean functions to construct a binary vector according to the feature values of **x**.

```
 1: procedure TREEDRIVENENCODER($\mathbf{x}$, $\mathcal{T}$))
 2:     $\mathbf{x}^b$ vector of length 0
 3:     for tree $T \in \mathcal{T}$ do
 4:         $\mathbf{u}$ vector of length $|T|$          ▷ binary vector we aim to construct
 5:         $i := 0$                        ▷ position index in the binary vector
 6:         $Q := \emptyset$ an empty queue
 7:         $Q$.enqueue(T.root)
 8:         while $Q$.notEmpty do
 9:             $v$ := Q.dequeue()
10:             $x$ := getFeatureValue($\mathbf{x}$, $v$)   ▷ get from x the value of the
    feature that is evaluated at v
11:             if $v$.evaluate($x$) == true then              ▷ evaluate x at v
12:                 add $\mathbf{u}(i) = 1$
13:             else
14:                 add $\mathbf{u}(i) = 0$
15:             end if
16:             i++
17:             for all children $v'$ of $v$ do
18:                 if $v'$ is an inner node ($v' \in V_I$) then
19:                     $Q$.enqueue($v'$)
20:                 end if
21:             end for
22:         end while
23:         $\mathbf{x}^b$ = concat($\mathbf{x}^b$, $\mathbf{u}$)
24:     end for
25:     return $\mathbf{x}^b$
26: end procedure
```

flexibility of the proposed framework allows to utilize almost any existing types of the DNNs.

3.4 DeepTLF and multimodal data

The proposed deep tabular learning framework can be employed for multimodal learning problems [18,46], where multimodal data involve both tabular and other data sources (e.g., text, image, or sound) in an integrated manner while achieving a robust performance. Our multimodal strategy is decoupled from any particular artificial neural network architecture, and thus, it can be easily integrated into an existing multimodal pipeline. Practically multimodal learning is done utilizing different data fusion strategies, i.e., early fusion, middle fusion, and late fusion [47–49].

In early fusion, input data samples can be directly concatenated. Formally, given two feature vectors from modalities I and II, $\boldsymbol{x}^{\mathrm{I}} \in \mathbb{R}^n$ and $\boldsymbol{x}^{\mathrm{II}} \in \mathbb{R}^m$, where n and m are numbers of variables in modalities I and II, respectively. Then, we can define the concatenation as $\mathbb{R}^n \oplus \mathbb{R}^m \mapsto \mathbb{R}^{n+m}$, by the map $(\boldsymbol{x}^{\mathrm{I}}, \boldsymbol{x}^{\mathrm{II}}) \mapsto (x_1^{\mathrm{I}}, \ldots, x_n^{\mathrm{I}}, x_1^{\mathrm{II}}, \ldots, x_m^{\mathrm{II}})$. The concatenation procedure can be accordingly further scaled for more then two modalities.

The middle fusion, sometimes also referred to as intermediate fusion in the literature, is typically used when data from different modalities come with different structures and dimensionalities which are homogeneous for each modality but heterogeneous across modalities, e.g., a multidimensional dataset with visual and audio data along with single-dimensional tabular data. Thus, it is challenging to directly concatenate the input data upfront. The middle fusion is done by utilizing the multi-input deep neural network architecture with two types of inputs: single-dimensional input (fully connected layer, recurrent layer, 1D CNN layer) or multi-dimensional layers (e.g., CNN layer). Then, the concatenation of the data signal can be done in the middle of the DNN. Similar to the middle fusion method, the late fusion combines data signals using the last layers.

The choice of the data fusion strategy depends on the modality of the dataset, downstream task, and hardware. In our experiments, we observe that the middle fusion performs better on the modalities. However, further research on data fusion is needed.

4 Experiments

To evaluate the performance of DeepTLF against state-of-the-art models, we employ eight real-world heterogeneous datasets of varying sizes from different application domains.

4.1 Experimental settings

4.1.1 Datasets

For the evaluation of DeepTLF, we used six heterogeneous and two multimodal dataset from different domains as described in Table 1; each dataset was previously featured in multiple published studies. The web access points and description of each dataset are in Appendix C.1. The data is preprocessed in the same way for each experiment; we do normalization and missing values subsection steps, except for GBDT and DeepTLF; since these approaches can handle missing values independently.

4.1.2 Baseline models

For the baseline models, we select the following algorithms: LR, linear or logistic regression models; k-Nearest Neighbors (kNN) [50] is a nonparametric machine learning method; Random Forest (RF) [13]; for GBDT [14], we utilize the XGBoost implementation [15]; DNN, A deep neural network with four fully connected layers and two DropOut layers [51]; Leafs+LR, A hybrid model, combining leaf index from a trained GBDT model and generalized linear models proposed in [25]; RLNs [26], Regularization Learning Networks

Table 1 Details of the datasets used in the experimental evaluations

	Dataset	#Samples	#Num	#Cat	Task
D1	HIGGS	11,000,000	28	0	Classification
D2	Default of clients	30,000	14	9	Classification
D3	Telecom churn	51,047	38	18	Classification
D4	Zillow	167,888	31	27	Regression
D5	Avocado prices	18,249	8	3	Regression
D6	California housing	20,640	8	0	Regression
D7	E-commerce clothing reviews	23,486	6	4	Classification
D8	PetFinder adoption prediction	14,993	3	14	Classification

#Sample is the number of data points, #Num is the number of numerical variables, and #Cat is the number of categorical variables in a dataset

(RLNs) is a dedicated to tabular learning DNN, which uses the counterfactual loss to tune its regularization hyperparameters efficiently. TabNet [19] is a deep tabular data learning architecture, which uses sequential attention to choose which features to reason from at each decision step; neural oblivious decision ensembles (NODE) [20] is a deep tabular data learning architecture, which generalizes ensembles of oblivious decision trees, but benefits from both end-to-end gradient-based optimization and the power of multilayer hierarchical representation learning; DeepGBM [24], a deep learning framework distilled by the GBDT algorithm; Net-DNF [8]; VIME [27], a self-supervised learning framework for tabular data; TabTransformer [22], a framework built using self-attention transformers; lastly, DeepTLF (*the proposed algorithm*), consisting of a four fully connected layers with the two DropOut layers to lower the overfitting effect, the full architecture is presented in Table 2. We *deliberately* select a relatively simple neural network model without advanced layers such as the batch normalization or attention (transformer) to demonstrate the power of our approach. By applying more sophisticated DL techniques, the model performance can be further improved.

Table 2 The DL architecture of the DeepTLF model

Layer	Parameters
Fully connected	#weights = 384
SWISH activation	
DropOut	$p = 0.23$
Fully connected	#weights = 64
SWISH activation	
DropOut	$p = 0.23$
Fully connected	#weights = 32
SWISH activation	
Fully connected	#weights = 1 or 2

4.2 Performance evaluation

Main benchmark In our performance evaluation, we partitioned each of the datasets using *(stratified) fivefold cross-validation*. Our quality measures are cross-entropy loss for classification and mean-squared error (MSE) for regression tasks. Results are reported in terms of mean and standard deviation values in Table 3. Furthermore, we conduct the 5×2 CV paired statistical t-test [52] to compare the proposed framework and GBDT model for all datasets from Table 3. Under the null hypothesis ($\mathbf{H}_0$) that the *GBDT and DeepTLF models have equal performance*, we also set the significance level to 0.05 ($\alpha = 0.05$), i.e., the critical region for a significant statistical difference between our model and the comparison methods. The results are presented in Table 4.

Corrupted data We also compare the performance of DeepTLF with a plain DNN and GBDT under corrupted data to verify the *robustness* of our deep tabular learning framework in scenarios of noisy labels, noisy data, and missing values in the training data (Fig. 3).

Noisy training data and labels We use two different setups: noisy training labels and noisy training data. We artificially corrupted the customer churn dataset by introducing random noise either to the training labels (labels were shuffled) and the training dataset. Note that for validation purposes, the test dataset was not corrupted. A distinguishing strength of the DeepTLF framework compared to other state-of-the-art approaches in the field is that it can handle missing values internally through the proposed gradient-boosting embeddings.

Missing values experiment Figure 9 in Appendix shows the performance of DNN, GBDT, and DeepTLF models with different proportions of missing in the training dataset. As we can see, the performance of the DNNs drops drastically, while DeepTLF shows stable performance.

Sensitivity to hyperparameters This experiment demonstrates how the GBDT hyperparameters contribute to the final performance of the DeepTLF such as the number of decision

Table 3 Experimental results based on (stratified) fivefold cross-validation

	Classification datasets			Regression datasets		
	D1	D2	D3	D4	D5	D6
LR	0.637 ± 0.001	0.470 ± 0.001	0.584 ± 0.001	0.028 ± 0.001	0.966 ± 0.001	0.552 ± 0.063
KNN [50]	Out-of-Memory	0.467 ± 0.003	0.600 ± 0.003	0.029 ± 0.001	0.038 ± 0.001	0.408 ± 0.011
RF [13]	0.502 ± 0.001	0.444 ± 0.007	0.564 ± 0.003	0.028 ± 0.001	0.025 ± 0.001	0.254 ± 0.008
GBDT [14]	0.498 ± 0.001	0.429 ± 0.006	0.559 ± 0.003	0.027 ± 0.003	0.026 ± 0.003	0.217 ± 0.021
Leafs+LR [25]	0.659 ± 0.001	0.453 ± 0.002	0.580 ± 0.002	0.029 ± 0.001	0.105 ± 0.036	0.358 ± 0.032
Deep neural network models						
DNN	0.511 ± 0.001	0.437 ± 0.005	0.579 ± 0.002	0.028 ± 0.001	0.069 ± 0.002	0.339 ± 0.122
RLN [26]	0.507 ± 0.002	0.433 ± 0.051	0.599 ± 0.001	0.399 ± 0.042	0.275 ± 0.244	0.947 ± 0.228
DeepGBM [24]	0.487 ± 0.001	0.457 ± 0.023	0.589 ± 0.001	**0.026 ± 0.001**	0.038 ± 0.045	0.299 ± 0.017
TabNet [19]	0.503 ± 0.001	0.447 ± 0.001	0.591 ± 0.005	0.049 ± 0.001	0.073 ± 0.002	0.455 ± 0.106
VIME [27]	0.514 ± 0.001	0.453 ± 0.006	0.593 ± 0.002	0.030 ± 0.003	0.120 ± 0.016	0.684 ± 0.023
TabTransformer [22]	0.581 ± 0.002	0.515 ± 0.003	0.650 ± 0.021	0.029 ± 0.001	0.073 ± 0.002	0.994 ± 0.501
Net-DNF [8]	0.561 ± 0.001	0.512 ± 0.001	0.594 ± 0.003	0.027 ± 0.001	0.321 ± 0.093	2.491 ± 0.051
NODE [20]	0.489 ± 0.006	0.458 ± 0.006	0.598 ± 0.001	0.028 ± 0.001	0.104 ± 0.030	0.722 ± 0.052
DeepTLF (ours)	**0.483 ± 0.001**	**0.427 ± 0.006**	**0.557 ± 0.003**	**0.026 ± 0.001**	**0.021 ± 0.005**	**0.215 ± 0.012**

We use the same fold splitting strategy for every dataset. The cross-entropy measure (lower is better) is selected for classification tasks and MSE measure (lower is better) is selected for regression problems, respectively. The top results for each dataset are marked in bold

Table 4 Results of the 5×2 CV paired statistical t-test between GBDT and DeepTLF models (where * means $p < 0.05$, ** means $p < 0.01$)

	D1	D2	D3	D4	D5	D6
p-value	0.022*	0.003**	0.027*	0.805	0.044*	0.002**

trees (Fig. 4) and learning rate (Fig. 10). For comparison purposes, we also add the GBDT baseline to the figures. It can be seen that DeepTLF does not require extensive hyperparameter tuning, since it reaches the saturation level.

t-SNE visualizations We also compare t-SNE visualizations [53] of the default of the clients dataset and a TreeDrivenEncoder encoded version of the same dataset; the results are shown in Fig. 5. It can be seen that TreeDrivenEncoder indeed preserves valuable information from the trained decision trees.

Multimodal data In this experiment, we demonstrate how the proposed framework performs on multimodal data. For that purpose, we select two multimodal datasets e-commerce clothing reviews [54] and PetFinder adoption prediction [55]. There e-commerce clothing reviews dataset consists of textual and tabular data modalities, and PetFinder adop-

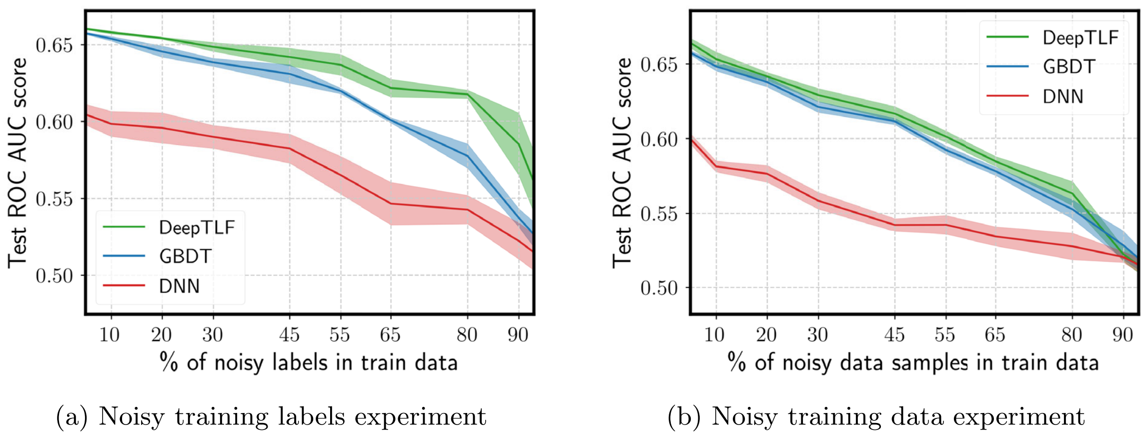


Fig. 3 The DNN here is identical to the DL part in the DeepTLF. Note the only the train data is corrupted, test data has the original values. We report the ROC AUC value (higher is better). Results are averages over five trials for the telecom churn (D3) dataset

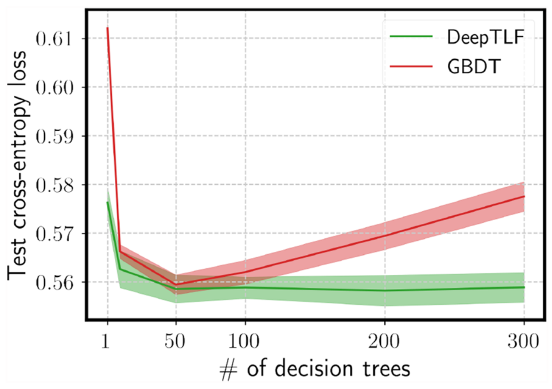


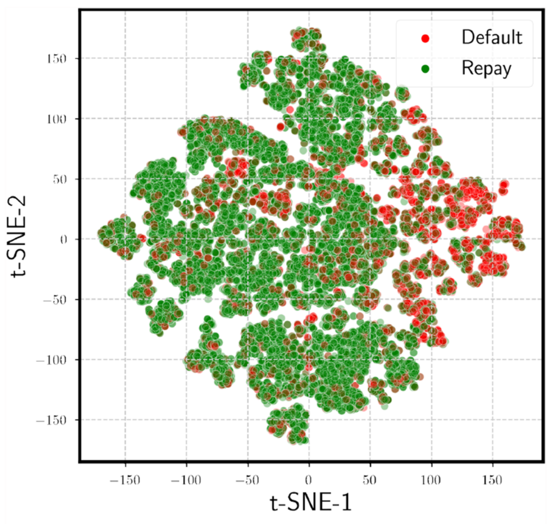
Fig. 4 A relationship between a number of trees in the GBDT and the final performance of the proposed DeepTLF framework. The precise same GBDT model is used for the data encoding in the DeepTLF. Results are averaged over five trials for the D3 dataset

tion prediction dataset has visual (images) and tabular data modalities. We compare DNN and DeepTLF models on unseen validation data (Fig. 6) using the middle-fusion strategy (Sect. 3.4) for both datasets. Tabular data representation is the only difference between DeepTLF and DNN baselines in this experiment, for DNN it is the original heterogeneous dataset after the normalization step, where the proposed framework utilizes TreeDrivenEncoder for the data transformation step. The results demonstrate the efficiency of our framework in the multimodal setting.

Training/Inference Runtime Comparison Finally, we compare the runtime performance between several DL-based algorithms with GBDT (XGBoost [15]). Table 5 summarizes our results. To make a fair comparison, we used the latest available versions of the corresponding implementations. Also, we utilize the same DL framework, PyTorch [56], and the same number of epochs as well as the batch size, for each DL-based baseline. One of the possible reasons for the gap between the proposed method and other DL-based approaches is that DeepTLF utilizes a simple deep neural network, whereas other approaches apply transformer networks or specialized decision tree-like layers. We also report the data preprocessing time for each baseline. The time cost of DeepTLF is increased compared with GBDT in the inference phase, due to the fact that the GBDT model is a well-optimized framework and written in C++, where DeepTLF is not yet fully optimized in terms of time efficiency and mostly written in Python; however, we do utilize the CUDA acceleration for training and inference steps.

5 Discussion

Empirical evaluations We can derive the following observations from experiments of the study: Our framework,



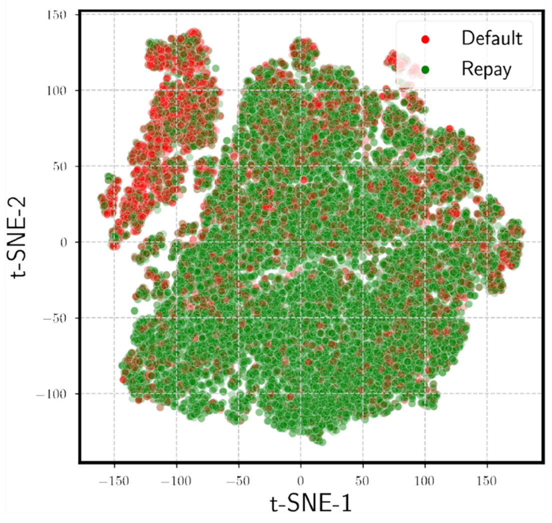


Fig. 5 t-SNE visualizations of original heterogeneous tabular default of clients dataset (top), and the same dataset after the TreeDrivenEncoder transformation (bottom)

DeepTFL, combines the preprocessing strengths of gradient-enhanced decision trees with the learning flexibility of deep neural networks. It can handle heterogeneity in the data very well and hence shows to be highly efficient. Also, the DeepTLF shows a stable performance irrespective of data size. On a large dataset, the DeepTLF approach demonstrates more than 3% improvement over the GBDT algorithm. We hypothesize that the improvement comes from the fact that deep neural networks perform better when a high number of data samples are available since DL models have more learnable parameters and, as a consequence, are more flexible than decision trees. Finally, with regard to data quality issues (noisy data and labels, missing values), our approach clearly outperforms the DNN and GBDT models, thus showing a robust performance under data quality challenges and

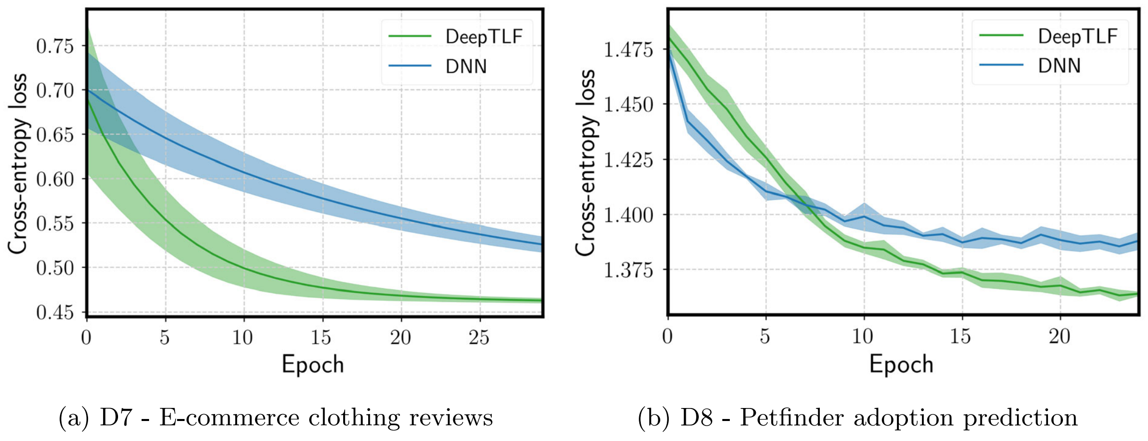


Fig. 6 We compare the performance of DNN and DeepTLF models using textual and tabular modalities from the D7 dataset and visual and tabular modalities from the D8 dataset, with identical DL architectures and training setups. The only difference between DeepTLF and DNN models in the experiment is the tabular data representation. Results are averaged over five trials

Table 5 A comparison of the training and inference runtime for selected models from different categories on the whole Zillow dataset (167,888 samples)

Model	Training time (s)	Inference time (s)	Data preprocessing time (s)	#Learning parameters
GBDT (CPU)	13.5	0.5	0	200 trees, depth 4
GBDT (GPU)	3.1	0.3	0	200 trees, depth 4
DNN (GPU)	10.1	0.64	0.3	53,551 weights
RLN (GPU)	10.4	2.22	0.3	60,355 weights
DeepGBM (GPU)	23.9	5.2	0.3	222,548 weights
TabNet (GPU)	79.1	2.2	0.3	584,832 weights
VIME (GPU)	30.2	5.5	0.3	99,732 weights
NODE (GPU)	310.2	15.5	0.3	27,105,922 weights
DeepTLF (GPU)	15.1	3.2	0	80,351 weights

The results related to the training, inference and preprocessing time are averages over five runs over the whole dataset for training and inference tests. The data preprocessing step includes: data scaling and handling missing values

it can *be applicable to many real-world applications where data loss occurs frequently.*

Decision tree model choice Noteworthy, the proposed prediction approach can use any decision tree ensemble as a basic algorithm; in this work, we adopt the GBDT method because of its well-known superior performance on heterogeneous tabular data and its robust feature handling capacities. In addition, the GBDT algorithm sequentially constructs the trees; at each step, the next tree maximally reduces the loss given the current loss. Thus, *there are conditional dependencies between the trees* in the GBDT ensemble, and as a consequence, they provide *adequate coverage of the data distribution*.

Hyperparameter selection for DeepTLF In our experiments, we demonstrate that the DeepTLF framework does not require extensive tinning for the decision tree ensemble part (Figs. 4 and 8); after reaching the saturation level, the number of trees does not have significant effect the performance of proposed framework.

Tabular data encoding Besides constructing a new homogeneous representation for the heterogeneous, tabular data, TreeDrivenEncoder encodes information about the whole dataset, as represented by the structures of the decision trees, which can be seen as a *local feature selection* (and feature engineering).

Furthermore, in terms of efficient representation, the encoded binary data has a drastically smaller size than the original heterogeneous data, since real-valued features are typically represented as 32-bit float types. In contrast, a binary vector can be efficiently represented by a sequence of Boolean values (i.e., 1 bit per value). *This allows for efficient training in the final component for the DeepTLF model.*

The TreeDrivenEncoder algorithm can be used for efficient categorical data encoding. In comparison with leaf-based encoding in DeepGBM [24,25], our transformation scheme utilizes the whole decision tree and produces binary features, whereas leaf-based encoding creates meta categorical features (indexing the leaves).

A comparison to Transformer-based models Most of the current state-of-the-art methods for deep learning on tabular data require an explicit definition of the categorical variables for a dataset, which might bring issues in the online setting, especially for environments with an increasing feature space. Moreover, a drawback of transformer-based methods is that the attention mechanism is applied to categorical values only, implying that the possible correlation between categorical and continuous variables is not taken into account. Furthermore, transformer-based approaches are learning representations for each category, which might be an issue for categorical variables with high cardinality. The proposed approach utilized the power of the decision trees encodes the all type of data together, therefore not suffering from aforementioned drawbacks.

Future work and limitations We see further potential in improving the efficiency of DeepTLF by replacing the decision trees with an efficient neural transformation layer, thus achieving an end-to-end deep learning mechanism for heterogeneous and multimodal data. However, with replacing the GBDT algorithm, the proposed framework loses the preprocessing powers essential for the tabular format [1]. Further improvements of our approach could be the usage of more advanced deep learning architectures such as convolution or attention-based neural networks [39].

Furthermore, an unsupervised training approach is desirable for the self-supervised learning techniques [57]. A possible way to do that is to use multiple variables as targets for the GBDT algorithm after the obtained feature vectors can be stacked into a single *meta* feature vector. Alternatively, the isolated forest algorithm [58] can be utilized for the first stage of the DeepTLF model. Also, the tabular data generation task is challenging due to its heterogeneous nature; however, with our proposed technique, which allows converting heterogeneous data into homogeneous, we see a lot of potentials.

Lastly, further analysis is needed to investigate the performance of DeepTLF in online learning scenarios. The goal would be to develop feature transformation mechanisms that can dynamically adjust to the data distribution's temporal changes and dimensionality. With regard to the GBDT algorithm, deep-learning-based algorithms allow efficient online training. However, DeepTLF works in a hybrid setting; therefore, for the next step, the gradient boosting decision trees might be replaced with a deep learning-based solution; this will lower the training and inference time.

6 Conclusion

In this work, we discussed the challenge of learning from heterogeneous tabular data with deep neural networks. The challenge stems from the concurrent existence of numerical and categorical feature types, complex, irregular dependencies between the features, and other data-related issues such as scales, outliers, and missing values. To address the challenge, we proposed DeepTLF, a framework that exploits the decision trees' structures from an ensemble model to map the original data into a homogeneous feature space where deep neural networks can be effectively and robustly trained. This allows DeepTLF to distill and conserve relevant information in the original data and utilize it in the deep-learning process. Furthermore, the distillation step reduces the required preprocessing to a minimum and can mitigate the mentioned data-related issues by exploiting decision trees' data-processing advantages (internal handling, missing values, and data scaling). Our extensive empirical evaluation on real-world datasets of different sizes and modalities convincingly showed that DeepTLF consistently outperforms the evaluated competitors, which are state-of-the-art approaches in this field. Also, the proposed framework showed robust performance on corrupted data (noisy labels, noisy data, and missing values). Compared to most approaches in this field, DeepTLF is easy to use and does not require changes to existing ML pipelines, which is essential for many practical applications. Moreover, we provide an open-source implementation of DeepTLF which can be used researchers and practitioners for various learning tasks on heterogeneous or multimodal tabular data.

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

Appendix A: Background: gradient boosting decision trees

Formally, at each iteration k of the gradient boosting algorithm, the GBDT model φ can be defined as:

$$\varphi^k(\mathbf{x}) = \varphi^{k-1}(\mathbf{x}) + \lambda\, h^k(\mathbf{x}), \tag{A1}$$

where $\mathbf{x}$ is an input feature vector, φ^{k-1} is the strong model constructed at the previous iteration, h is a weak learner from a family of functions $\mathcal{H}$, and λ is the learning rate.

$$h^k = \arg\min_{h \in \mathcal{H}} \sum_i \left(-\frac{\partial L(\varphi^{k-1}(\mathbf{x}_i), y_i)}{\partial \varphi^{k-1}(\mathbf{x}_i)} - h(\mathbf{x}_i) \right)^2 . \tag{A2}$$

More specifically, a pseudo-residual $-\frac{\partial L(\varphi^{k-1}(\mathbf{x}_i), y_i)}{\partial \varphi^{k-1}(\mathbf{x}_i)}$ should be approximated as well as possible by the current weak model $h(\mathbf{x}_i)$. The gradient w.r.t. the current predictions indicates how these predictions should be changed in order to minimize the loss function. Informally, gradient boosting can be thought of as performing gradient descent in the functional space.

Appendix B: Additional experiments

In the following section, we provide further experimental results to support the proposed deep neural network model on heterogeneous tabular data.

Validation loss curves We examine DNNs and our DeepTLF model separately using only the validation (unseen) data. To enable a fair comparison, the deep learning part in DeepTLF is identical to the DNN we used. The results are presented in Fig. 7.

Is there a correlation between GBDT's performance and the final performance of the DeepTLF? In this experiment, we examine the correlation between the performance of GBDT (which is used for data encoding) and DeepTLF (Figs. 8, 9 and 10). In other words, we want to demonstrate that if the performance of the GBDT improves, the performance of the DeepTLF rises. Figure 11 presents the results of the experiments; as it can be observed, there is indeed a high positive correlation between the performance of the GBDT and DeepTLF. It is also noticeable that the DeepTLF performance in many cased the GBDT results.

A "sanity check" experiment In this simple experiment, we want to verify that the proposed encoding function distills knowledge better than a random "encoding function". In order to do so, we design a random function such that it extracts a random feature f with random splitting value. The experiment confirms that indeed the proposed encoder performs better than a random set of Boolean rules for a given dataset (Fig. 12).

Appendix C: Reproducibility details

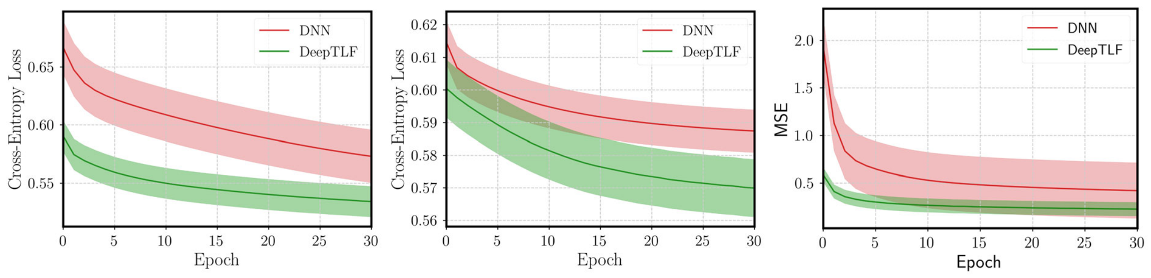
We use the following implementation of baseline ML models: kNN, RF, LR, algorithms are from the widely used open-source machine learning python library Scikit-Learn [59], for GBDT we select the python version of distributed gradient boosting library XGBoost [15], RLN, we use the official TensorFlow implementation from the GitHub repository[2], we use well-tested PyTorch implementation of TabNet[3]. We use the official implementation of Net-DNF[4]. We use the official PyTorch implementation of NODE from the GitHub repository[5], we adapt the official implementation of DeepGBM[6] to our need using the PyTorch framework. VIME[7] and TabTansformer[8] implementations are from official GitHub repositories. DeepTLF, for the encoding part, we employ the XGBoost[9] implementation of the GBDT algorithm; for the deep learning part, PyTorch [56] is used. Additionally, we will provide a TensorFlow implementation. Note, other implementations of GBDT can be used as well. We select the AdaBelief Optimizer [60] for proposed framework.

The DNN architecture Table 2 presents the architecture of the DL part of the DeepTLF model, which is identical to the architecture of the DNN baseline. We utilize the three common used neural network layers: fully connected (dense), DropOut [51], and activation layers. For activation, we select the SWISH function proposed in [61]. For the hyper-parameter selection task for all baseline, we apply the tree-structured parzen estimator (TPE) optimization algorithm using the HyperOpt library [62].

t-SNE Experiment For the t-SNE experiments, we scaled the original datasets by applying the z-score normalization. We did not preprocessed an encoded homogeneous dataset after the TreeDrivenEncoder transformation.

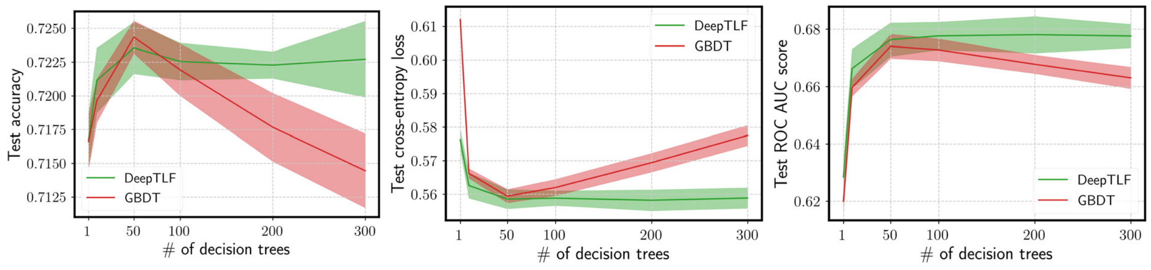
Computing infrastructure Out experimental setup for all experiments has two RTX2080Ti GPUs and a single CPU AMD 3960X 24-Core with the Ubuntu 20.04 operation system.

[2] https://github.com/irashavitt/regularization_learning_networks
[3] https://github.com/dreamquark-ai/tabnet
[4] https://github.com/amramabutbul/DisjunctiveNormalFormNet
[5] https://github.com/Qwicen/node
[6] https://github.com/motefly/DeepGBM
[7] https://github.com/jsyoon0823/VIME
[8] https://github.com/lucidrains/tab-transformer-pytorch
[9] https://xgboost.readthedocs.io/en/latest/



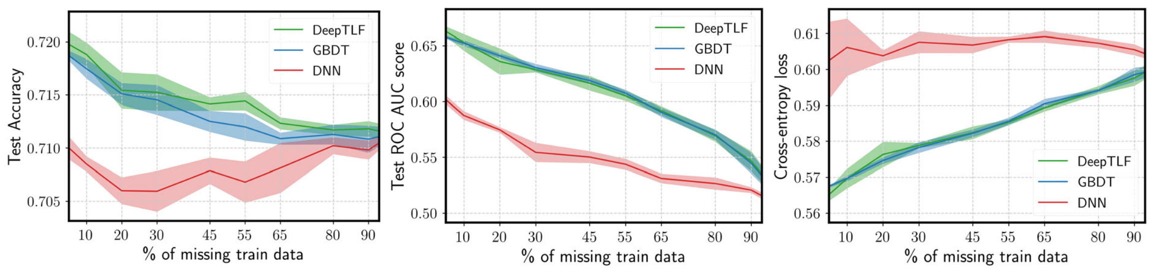
(a) D1 - HIGGS dataset (b) D3 - Telecom churn dataset (c) D6 - California housing dataset

Fig. 7 The comparison of the DeepTLF (a green line) and the deep neural model (DNN) (a red line) models on validation (unseen) data. DeepTLF and DNN models have the exact same architecture. The results are computed over ten runs with different random seeds (colour figure online)



(a) Accuracy score (b) ROC AUC score (c) Cross-entopy loss

Fig. 8 A relationship between number of decision trees and the DeepTLF performance. The accuracy score (higher is better), ROC AUC score (higher is better), cross-entropy loss (lower is better) metrics for the same experiment. The exact same GBDT model is used for the data encoding in the DeepTLF. The results are averages over five trials for the telecom churn (D3) dataset

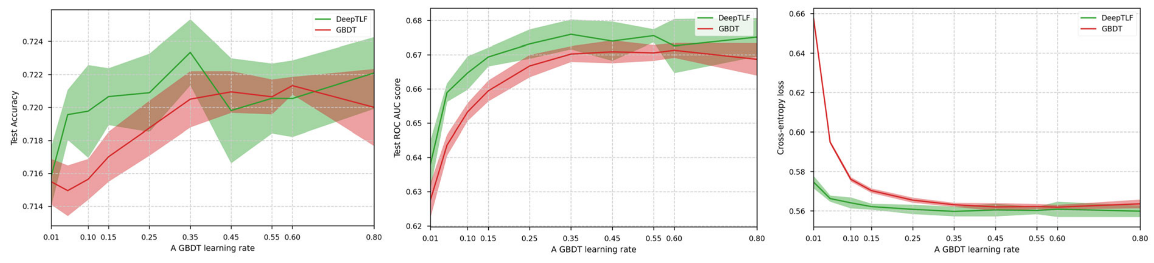


(a) Accuracy score (b) ROC AUC score (c) Cross-entopy loss

Fig. 9 The missing data experiment. The accuracy score (higher is better), ROC AUC score (higher is better), cross-entopy loss (lower is better) metrics for the same experiment. The exact same GBDT model is used for the data encoding in the DeepTLF. The DNN model is identical in training and architecture to the DeepTLF's DNN part. The results are averages over five trials for the telecom churn (D3) dataset

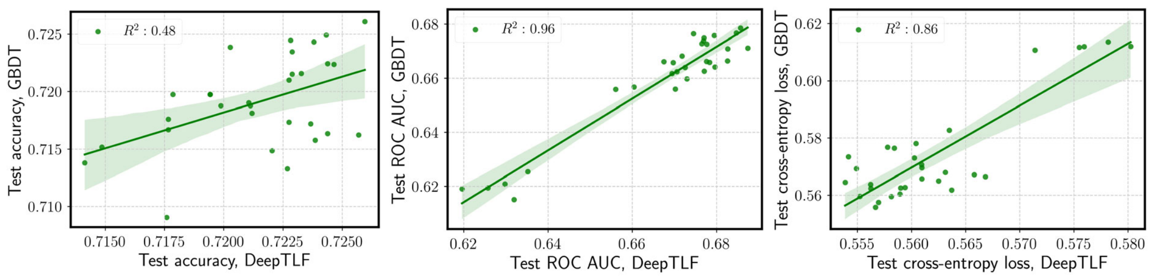
C.1: Datasets description

Among these, the *HIGGS* dataset, which stems from experimental physics, is the largest dataset in our evaluation. As an exemplary dataset from the financial industry, we include the dataset *defaults of clients*, which contains information on default payments, demographic factors, credit data, history of payment, and bill Statements of credit card clients in Taiwan from April 2005 to September 2005. In addition, the *Zillow* dataset represents typical heterogeneous data from the real estate sector. It is important to emphasize that in this dataset around 47 % of the data inputs are missing values. The avocado dataset is another representative of tabular datasets, which provides historical data on avocado prices. The *telecom churn* dataset presents customer data of different feature types with the goal to estimate the behavior of a customer. The *California housing* dataset which contains information about house pricing in 1990. Lastly, we employ two mul-



(a) Accuracy score (b) ROC AUC score (c) Cross-entopy loss

Fig. 10 A relationship between the GBDT learning rate and the DeepTLF performance. In this experiment, we want to show the performance changes of the DeepTLF model by varying the learning rate parameter in the GBDT algorithm. The results are averaged over five trials for the telecom churn (D3) dataset



(a) Accuracy score (b) ROC AUC score (c) Cross-entopy loss

Fig. 11 Correlation plots for different quality measurements. The exact same GBDT model is used for the data encoding in the DeepTLF. The results demonstrate that there is indeed a high positive relationship between the performance of GBDT and DeepTLF. Thus, the proposed data distillation algorithm can successfully distill the knowledge from trees. The results are averaged over five trials for the telecom churn (D3) dataset

timodal datasets: *E-commerce clothing reviews* dataset [54] with text and tabular data, and the *PetFinder adoption prediction* dataset [55] with visual and tabular data, it consists of information on cats and dogs with associated images.

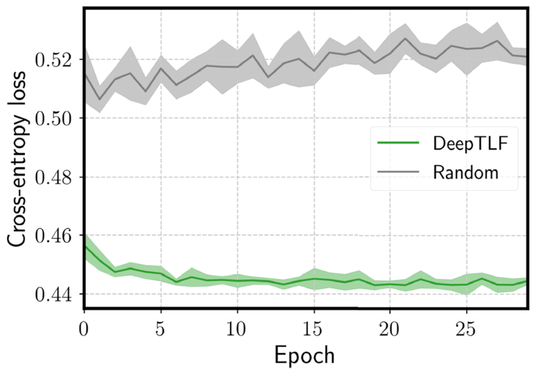


Fig. 12 A "sanity check" experiment. A comparison of the TreeDrivenEncoder and random encoding functions. The random encoding function mimics the TreeDrivenEncoder, but it selects a random feature and splitting value. The experiment verifies that the TreeDrivenEncoder is able to distill the knowledge using trained decision trees in a GBDT algorithm. The results are averaged over five trials for the telecom churn (D3) dataset

All these datasets are collected from real-world problems and contain numerical as well as categorical data. Moreover, these datasets are freely available online and common in tabular data processing: each dataset was previously featured in multiple published studies. We deliberately chose these eight datasets to cover different domain areas (web, natural sciences, etc.), tasks (classification and regression), different dataset sizes, and various data modalities.

Table 6 presents positive and negative class ratios for the classification datasets of this study. The online links to each dataset are provided in Table 7.

We prepossessed the data in the same way for every baseline model by applying standard normalization. For the linear regression, logistic regression, and models based on neural networks, the missing values were substituted with zeros since these methods cannot handle them otherwise.

Appendix A Appendix

Table 6 Positive and negative class ratios for binary classification datasets used in the study

	Dataset	Negative class (0) %	Positive class (1) %
D1	HIGGS	0.47	0.53
D2	Default of Clients	0.778	0.221
D3	Telecom churn	0.712	0.288
D4	E-commerce clothing reviews	0.178	0.822

Table 7 URLs for datasets of the study

	Dataset	URL
D1	HIGGS	https://archive.ics.uci.edu/ml/datasets/HIGGS
D2	Default of clients	https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset
D3	Telecom churn	https://www.kaggle.com/c/zillow-prize-1
D4	Zillow	https://www.kaggle.com/neuromusic/avocado-prices
D5	Avocado prices	https://www.kaggle.com/blastchar/telco-customer-churn
D6	California housing	https://www.kaggle.com/camnugent/california-housing-prices
D7	E-commerce clothing reviews	https://www.kaggle.com/nicapotato/womens-ecommerce-clothing-reviews
D8	PetFinder adoption prediction	https://www.kaggle.com/competitions/petfinder-adoption-prediction/data

References

1. Borisov, V., Leemann, T., Seßler, K., Haug, J., Pawelczyk, M., Kasneci, G.: Deep neural networks and tabular data: a survey. arXiv preprint arXiv:2110.01889 (2021)
2. Fatima, M., Pasha, M., et al.: Survey of machine learning algorithms for disease diagnostic. J. Intell. Learn. Syst. Appl. **9**(01), 1 (2017)
3. Dastile, X., Celik, T., Potsane, M.: Statistical and machine learning models in credit scoring: a systematic literature survey. Appl. Soft Comput. **91**, 106263 (2020)
4. Buczak, A.L., Guven, E.: A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Commun. Surv. Tutor. **18**(2), 1153–1176 (2015)
5. Goodfellow, I., Bengio, Y., Courville, A.: Deep learning (2016). http://www.deeplearningbook.org
6. Shwartz-Ziv, R., Armon, A.: Tabular data: deep learning is not all you need (2021)
7. Mitchell, B.R., et al.: The spatial inductive bias of deep learning. PhD thesis, Johns Hopkins University (2017)
8. Katzir, L., Elidan, G., El-Yaniv, R.: Net-DNF: effective deep modeling of tabular data. In: International Conference on Learning Representations (2020)
9. García, S., Luengo, J., Herrera, F.: Data preprocessing in data mining, vol. 72. Springer, Cham, Switzerland (2015)
10. Hancock, J.T., Khoshgoftaar, T.M.: Survey on categorical data for neural networks. J. Big Data **7**, 1–41 (2020)
11. Gorishniy, Y., Rubachev, I., Babenko, A.: On embeddings for numerical features in tabular deep learning. arXiv preprint arXiv:2203.05556 (2022)
12. Nielsen, D.: Tree boosting with xgboost-why does xgboost win "every" machine learning competition? Master's thesis, NTNU (2016)
13. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
14. Friedman, J.H.: Stochastic gradient boosting. Comput. Stat. Data Anal. **38**(4), 367–378 (2002)
15. Chen, T., Guestrin, C.: XGBoost: a scalable tree boosting system. In: Proceedings of the 22nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining, pp. 785–794 (2016)
16. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.-Y.: Lightgbm: a highly efficient gradient boosting decision tree. In: Advances in Neural Information Processing Systems, pp. 3146–3154 (2017)
17. Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A.V., Gulin, A.: CatBoost: unbiased boosting with categorical features. In: Advances in Neural Information Processing Systems, pp. 6638–6648 (2018)
18. Gu, K., Budhkar, A.: A package for learning on tabular and text data with transformers. In: Proceedings of the Third Workshop on Multimodal Artificial Intelligence, pp. 69–73 (2021)
19. Arik, S.O., Pfister, T.: TabNet: attentive interpretable tabular learning. arXiv preprint arXiv:1908.07442 (2019)
20. Popov, S., Morozov, S., Babenko, A.: Neural oblivious decision ensembles for deep learning on tabular data. arXiv preprint arXiv:1909.06312 (2019)
21. Yin, P., Neubig, G., Yih, W.-T., Riedel, S.: Tabert: pretraining for joint understanding of textual and tabular data. arXiv preprint arXiv:2005.08314 (2020)
22. Huang, X., Khetan, A., Cvitkovic, M., Karnin, Z.: Tabtransformer: tabular data modeling using contextual embeddings. arXiv preprint arXiv:2012.06678 (2020)
23. Guo, H., Tang, R., Ye, Y., Li, Z., He, X.: DeepFM: a factorization-machine based neural network for CTR prediction. arXiv preprint arXiv:1703.04247 (2017)
24. Ke, G., Xu, Z., Zhang, J., Bian, J., Liu, T.-Y.: DeepGBM: a deep learning framework distilled by GBDT for online prediction tasks. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 384–394 (2019)
25. He, X., Pan, J., Jin, O., Xu, T., Liu, B., Xu, T., Shi, Y., Atallah, A., Herbrich, R., Bowers, S., et al.: Practical lessons from predicting clicks on ads at facebook. In: Proceedings of the Eighth International Workshop on Data Mining for Online Advertising, pp. 1–9 (2014)
26. Shavitt, I., Segal, E.: Regularization learning networks: deep learning for tabular datasets. In: Advances in Neural Information Processing Systems, pp. 1379–1389 (2018)
27. Yoon, J., Zhang, Y., Jordon, J., van der Schaar, M.: Vime: extending the success of self-and semi-supervised learning to tabular domaindim. Adv. Neural Inf. Process. Syst. **33**, 11033–11043 (2020)

28. Padhi, I., Schiff, Y., Melnyk, I., Rigotti, M., Mroueh, Y., Dognin, P., Ross, J., Nair, R., Altman, E.: Tabular transformers for modeling multivariate time series. arXiv preprint arXiv:2011.01843 (2020)
29. Levy, E., Mathov, Y., Katzir, Z., Shabtai, A., Elovici, Y.: Not all datasets are born equal: on heterogeneous data and adversarial examples. arXiv preprint arXiv:2010.03180 (2020)
30. Ballet, V., Renard, X., Aigrain, J., Laugel, T., Frossard, P., Detyniecki, M.: Imperceptible adversarial attacks on tabular data. arXiv preprint arXiv:1911.03274 (2019)
31. Akrami, H., Aydore, S., Leahy, R.M., Joshi, A.A.: Robust variational autoencoder for tabular data with beta divergence. arXiv preprint arXiv:2006.08204 (2020)
32. Gupta, K., Pesquet-Popescu, B., Kaakai, F., Pesquet, J.-C.: A quantitative analysis of the robustness of neural networks for tabular data. In: ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 8057–8061 (2021). IEEE
33. Rendle, S.: Factorization machines. In: 2010 IEEE International Conference on Data Mining, pp. 995–1000 (2010). IEEE
34. Lian, J., Zhou, X., Zhang, F., Chen, Z., Xie, X., Sun, G.: xDeepFM: combining explicit and implicit feature interactions for recommender systems. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1754–1763 (2018)
35. Rota Bulo, S., Kontschieder, P.: Neural decision forests for semantic image labelling. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 81–88 (2014)
36. Denoyer, L., Gallinari, P.: Deep sequential neural network. arXiv preprint arXiv:1410.0510 (2014)
37. Wang, S., Aggarwal, C., Liu, H.: Using a random forest to inspire a neural network and improving on it. In: Proceedings of the 2017 SIAM International Conference on Data Mining, pp. 1–9 (2017). SIAM
38. Peters, B., Niculae, V., Martins, A.F.: Sparse sequence-to-sequence models. arXiv preprint arXiv:1905.05702 (2019)
39. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. arXiv preprint arXiv:1706.03762 (2017)
40. Baylor, D., Breck, E., Cheng, H.-T., Fiedel, N., Foo, C.Y., Haque, Z., Haykal, S., Ispir, M., Jain, V., Koc, L., et al.: Tfx: a tensorflow-based production-scale machine learning platform. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1387–1395 (2017)
41. Moosmann, F., Triggs, B., Jurie, F.: Fast discriminative visual codebooks using randomized clustering forests. In: Advances in Neural Information Processing Systems, pp. 985–992 (2007)
42. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Mach. Learn. **63**(1), 3–42 (2006)
43. Medvedev, D., D'yakonov, A.: New properties of the data distillation method when working with tabular data. arXiv preprint arXiv:2010.09839 (2020)
44. Bruch, S., Pfeifer, J., Guillame-bert, M.: Learning representations for axis-aligned decision forests through input perturbation. arXiv preprint arXiv:2007.14761 (2020)
45. Pedapati, T., Balakrishnan, A., Shanmugam, K., Dhurandhar, A.: Learning global transparent models consistent with local contrastive explanations. Adv. Neural Inf. Process. Syst. **33**, 3592–3602 (2020)
46. Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H., Ng, A.Y.: Multimodal deep learning. In: ICML (2011)
47. Boulahia, S.Y., Amamra, A., Madi, M.R., Daikh, S.: Early, intermediate and late fusion strategies for robust deep learning-based multimodal action recognition. Mach. Vis. Appl. **32**(6), 1–18 (2021)
48. Ma, M., Ren, J., Zhao, L., Testuggine, D., Peng, X.: Are multimodal transformers robust to missing modality? In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 18177–18186 (2022)
49. Nagrani, A., Yang, S., Arnab, A., Jansen, A., Schmid, C., Sun, C.: Attention bottlenecks for multimodal fusion. Adv. Neural Inf. Process. Syst. **34**, 14200–14213 (2021)
50. Fix, E.: Discriminatory analysis: nonparametric discrimination, consistency properties (1951)
51. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(56), 1929–1958 (2014)
52. Dietterich, T.G.: Approximate statistical tests for comparing supervised classification learning algorithms. Neural Comput. **10**(7), 1895–1923 (1998)
53. Van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. J. Mach. Learn. Res. **9**(11), 2579–2605 (2008)
54. Brooks, N.: Women's E-commerce clothing reviews. Data retrieved from Kaggle, https://www.kaggle.com/datasets/nicapotato/womens-ecommerce-clothing-reviews (2018)
55. PetFinder.my: PetFinder.my adoption prediction. data retrieved from Kaggle, https://www.kaggle.com/competitions/petfinder-adoption-prediction (2019)
56. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch. In: NIPS-W (2017)
57. Jaiswal, A., Babu, A.R., Zadeh, M.Z., Banerjee, D., Makedon, F.: A survey on contrastive self-supervised learning. Technologies **9**(1), 2 (2021)
58. Liu, F.T., Ting, K.M., Zhou, Z.-H.: Isolation forest. In: 2008 Eighth IEEE International Conference on Data Mining, pp. 413–422 (2008). IEEE
59. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: machine learning in python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
60. Zhuang, J., Tang, T., Ding, Y., Tatikonda, S., Dvornek, N., Papademetris, X., Duncan, J.S.: Adabelief optimizer: adapting stepsizes by the belief in observed gradients. arXiv preprint arXiv:2010.07468 (2020)
61. Ramachandran, P., Zoph, B., Le, Q.V.: Searching for activation functions. arXiv preprint arXiv:1710.05941 (2017)
62. Bergstra, J., Bardenet, R., Bengio, Y., Kégl, B.: Algorithms for hyper-parameter optimization. Neural Inf. Process. Syst. Found. **24**, 2546–2554 (2011)

A.1.3 Language Models are Realistic Tabular Data Generators

Publication: Published at the Eleventh International Conference on Learning Representations (ICLR), 2023.

Contribution: Kathrin Seßler and I were the primary creators of the GReaT framework. Additionally, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and I also performed the experiments, where I wrote most parts of the paper. Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci provided valuable input and feedback throughout the process, helping to refine and improve the ideas, formalizations, and analysis in the paper. All co-authors played a role in reviewing and revising the final manuscript.

Language Models are Realistic Tabular Data Generators

Vadim Borisov[1]*, **Kathrin Seßler**[2]*, **Tobias Leemann**[1], **Martin Pawelczyk**[1], **Gjergji Kasneci**[1]
[1]University of Tübingen, Tübingen, Germany
[2]Technical University of Munich, Munich, Germany

Abstract

Tabular data is among the oldest and most ubiquitous forms of data. However, the generation of synthetic samples with the original data's characteristics still remains a significant challenge for tabular data. While many generative models from the computer vision domain, such as autoencoders or generative adversarial networks, have been adapted for tabular data generation, less research has been directed towards recent transformer-based large language models (LLMs), which are also generative in nature. To this end, we propose **GReaT** (**G**eneration of **Rea**listic **T**abular data), which exploits an auto-regressive generative LLM to sample synthetic and yet highly realistic tabular data. Furthermore, GReaT can model tabular data distributions by conditioning on any subset of features; the remaining features are sampled without additional overhead. We demonstrate the effectiveness of the proposed approach in a series of experiments that quantify the validity and quality of the produced data samples from multiple angles. We find that GReaT maintains state-of-the-art performance across many real-world data sets with heterogeneous feature types.

tl&dr The proposed GReaT approach utilizes the capabilities of pretrained large language models to synthesize realistic tabular data. A challenging set of experiments validates the GReaT method's high generative qualtity.

1 Introduction

Tabular data is one of the most common forms of data in machine learning (ML) – over 65% of data sets in the Google Dataset Search platform contain tabular files in either CSV or XLS formats (Benjelloun et al., 2020). However, due to the expensive nature of data collection processes, tabular data sets (i) are often class imbalanced, (i.e., tabular data sets tend to have long-tailed label distributions (Cao et al., 2019)), (ii) contain critical person-related information and cannot be shared due to privacy protection or socio-ethical principles (Gascón et al., 2017), and (iii) often come with impurity issues such as noisy or missing values which impede the application of modern ML algorithms (Lin & Tsai, 2020). Synthetically generated data has the potential to alleviate these three important issues. Therefore, the generation of realistic artificial tabular data has received considerable attention in recent years (Choi et al., 2017; Park et al., 2018; Xu et al., 2019; Borisov et al., 2021).

Apart from real-world impurity issues, there also exist various technical problems that make the generation of synthetic data difficult. Typically, tabular data contains various feature types, such as *categorical features* (e.g., `name`, `countryOfOrigin`, `jobTitle`) and *numerical features* (e.g., `age`, `income`), and can be easily visualized in a table (e.g., see upper left panel in Figure 2). The categorical variables (i.e., words or clauses) may frequently contain most of the information. For example, the highly used Adult Income data set consists of seven numerical and eight categorical variables (Dua & Graff, 2017). This heterogeneity of feature types and values leads to three core challenges in tabular data preprocessing and analysis:

Extensive and lossy preprocessing. For most of the existing tabular data generation methods, extensive data preprocessing of tabular data is required, which usually includes the following steps:

*Equal contribution
Corresponding authors: kathrin.sessler@tum.de, vadim.borisov@uni-tuebingen.de

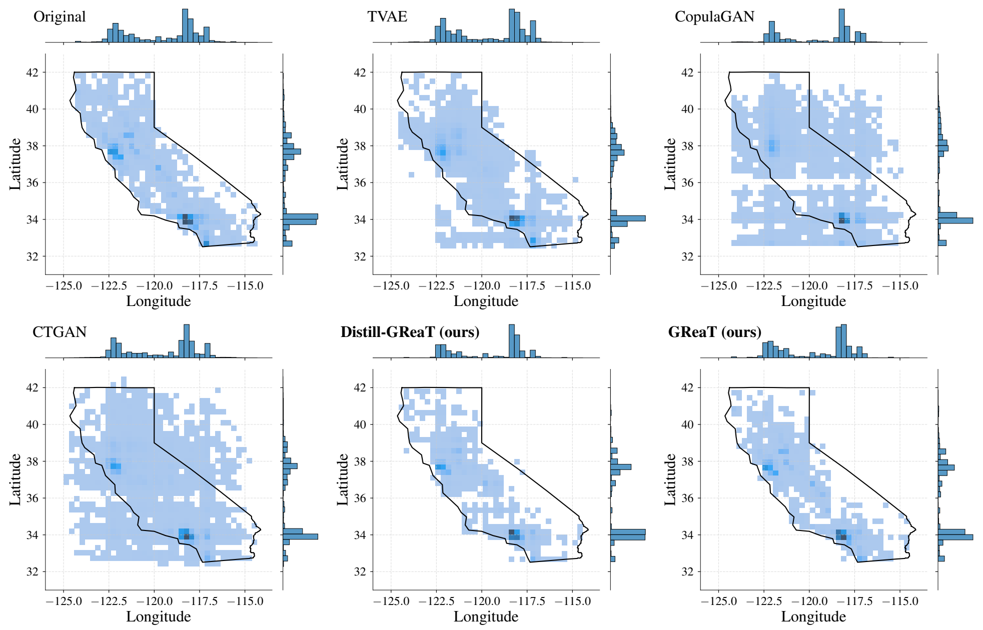


Figure 1: A comparison of the original and generated samples for the California Housing data set (Pace & Barry, 1997), which contains characteristic information about different properties in California, USA. We show joint histogram plots of the highly interconnected variables `Latitude` and `Longitude`. The black outline indicates the true boundary of the state of California.

(i) categorical data encoding into numbers, (ii) data scaling or normalization, (iii) replacing missing values, and (iv) removing outliers and smoothing. These data transformation steps may result in the loss of important information or the introduction of artifacts that are not present in the original data. As an example, the categorical encoding into numeric values may introduce an artificial ordering into previously unordered values (Borisov et al., 2021). Therefore, the problem of *lossy preprocessing* may strongly influence the quality of generated data as a result.

Context knowledge for coherent semantics. Almost all common synthetic data generation methods transform tabular data into a fully numerical representation. However, tabular data sets frequently consist of variables that are contextually interconnected. In the Adult Income data set (Dua & Graff, 2017), the features `age`, `marital-status`, and `education` have a clear coherence relationship: There is a certain minimal legal age for marriage, and it is challenging to get a Ph.D. at a young age. Such context knowledge should ideally be considered when generating *realistic* synthetic data samples. We refer to this common issue as the *contextual knowledge problem*.

Arbitrary conditioning. A versatile model that can generate data for a large variety of applications should be able to synthesize data conditioned on an arbitrary set of variables. This allows imputation of any missingness pattern in the data or oversampling of arbitrary subsets. Currently, the majority of the methods do not provide extensions to arbitrary conditioning and require the generative model to be re-trained according to each specific set of features to be conditioned on (Mirza & Osindero, 2014). We refer to generators that allow for conditional generation with any specified feature combination as supporting *arbitrary conditioning*.

Most modern deep-learning approaches for tabular data generation built on generative models transferred from the computer vision domain (Borisov et al., 2021), such as Variational Autoencoders (VAEs, Kingma & Welling, 2013) or Generative Adversarial Networks (GANs, Goodfellow et al., 2014). However, deep learning models have equally revolutionized the field of natural language processing (NLP, Radford et al., 2019; Brown et al., 2020; Raffel et al., 2020). Modern large language models (LLMs) are often constructed in the form of auto-regressive density models over sequences of words (Radford et al., 2019; Bengio et al., 2000). This begs the question to which extent successful architectures for NLP are apt to the tabular data generation task.

Carrying this thought further, we present a novel method for probabilistic data generation that covers the outlined core challenges and results in state-of-the-art performance (see Fig. 1 for a qualitative example). We argue that pretrained self-attention-based LLMs (Vaswani et al., 2017) are suitable for the probabilistic modeling of heterogeneous tabular data sets after these data sets have been appropriately transformed into a *textual representation*. We do so by constructing syntactically correct sentences based on feature names and row values without losing information or introducing artificial orderings and thus mitigate the issue of *lossy preprocessing*. This step, to which we refer as *textual encoding*, maintains substantially more information than usual transformations. Since we include the variable names in the encoding, a model trained on this data can directly access contextual information. To be able to make sense of this information, we suggest using established and pretrained language models to perform the generation task. This could be a possible path towards tackling the *contextualization problem*. Finally, we introduce a feature order permutation step to shuffle the feature order in the textual encodings. Training a model on such data will result in a versatile generator that supports *arbitrary conditioning*. Specifically, our work offers the following contributions relative to existing literature on the generation of synthetic tabular data:

- **Novel paradigm.** We propose the first approach for realistic heterogeneous tabular data modeling and generation utilizing a transformer-decoder network architecture. Thereby, *we connect tabular and textual data modalities via a textual encoding scheme*.
- **Arbitrary conditioning.** When trained on textual encodings with random feature order permutations, our model inherits its arbitrary conditioning power from the LLM, which can model the data distribution conditioned on any given subset of features and sample the remaining features
- **Extensive experimental results.** We show that our **G**eneration of **Rea**listic **T**abular Data (GReaT) obtains state-of-the-art generative performance on a variety of data sets across several measures. We have open-sourced our experimental results, making them available as strong benchmarks for the benefit of the community.
- **Python package.** Finally, we provide an easy-to-use Python implementation of the GReaT model, where it takes only three lines of code to generate new synthetic samples. The code is accessible via `pip install be-great`.[1]

2 Related Work

While the generation of images and text is extensively explored (Karras et al., 2020; Subramanian et al., 2017), the generation of synthetic tabular is less commonly considered in the recent machine learning literature. Classical methods for tabular data modeling include Bayesian networks, in particular those based on the Chow-Liu approximation (Chow & Liu, 1968) or statistical tools such as copulas (Kamthe et al., 2021). More recent methods for probabilistic tabular data modeling utilize generative adversarial networks (Choi et al., 2017; Park et al., 2018; Mottini et al., 2018; Xu et al., 2019; Koivu et al., 2020) or variational autoencoders (Xu et al., 2019; Ma et al., 2020; Vardhan & Kok, 2020; Darabi & Elor, 2021). A related line of work considers the synthetic generation of multi-variate time series data (Padhi et al., 2021).

The mixed structure of discrete and continuous features along with their different value distributions still poses a significant challenge. CTGAN, the current state-of-the-art approach by Xu et al. (2019), places special focus on the conditional distributions between the features to generate semantically meaningful data. For non-Gaussian feature distributions, the authors propose a *mode-specific normalization* technique. However, as we pointed out in Section 1, such encoding techniques can be lossy and introduce artificial orderings. Furthermore, they do not profit from contextual information. We thoroughly compare our method to their approach in the experimental evaluation section.

In a parallel development, the area of natural language processing was dominated by recurrent neural networks (RNNs) before self-attention-based neural networks (Vaswani et al., 2017) revolutionized this field. Based on their technique, various auto-encoding models (Devlin et al., 2018; Sanh et al., 2019; Lan et al., 2019) for tasks like sentence classification, sequence-to-sequence models (Raffel et al., 2020; Lewis et al., 2020) for translation or summarizing, and auto-regressive models (Radford

[1] `https://github.com/kathrinse/be_great`

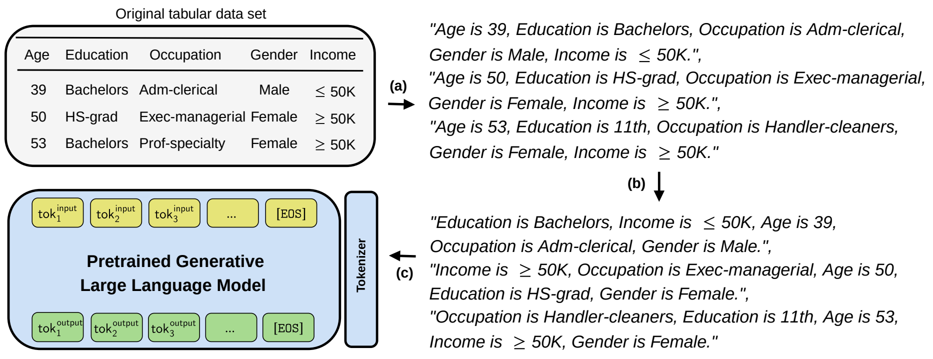

Original tabular data set
Age Education Occupation Gender Income
39 Bachelors Adm-clerical Male ≤ 50K
50 HS-grad Exec-managerial Female ≥ 50K
53 Bachelors Prof-specialty Female ≥ 50K
(a)
"Age is 39, Education is Bachelors, Occupation is Adm-clerical, Gender is Male, Income is ≤ 50K.",
"Age is 50, Education is HS-grad, Occupation is Exec-managerial, Gender is Female, Income is ≥ 50K.",
"Age is 53, Education is 11th, Occupation is Handler-cleaners, Gender is Female, Income is ≥ 50K."
(b)
"Education is Bachelors, Income is ≤ 50K, Age is 39, Occupation is Adm-clerical, Gender is Male.",
"Income is ≥ 50K, Occupation is Exec-managerial, Age is 50, Education is HS-grad, Gender is Female.",
"Occupation is Handler-cleaners, Education is 11th, Age is 53, Income is ≥ 50K, Gender is Female."
(c)
Tokenizer
Pretrained Generative Large Language Model
tok input 1
tok input 2
tok input 3
...
[EOS]
tok output 1
tok output 2
tok output 3
...
[EOS]

Figure 2: The GReaT data pipeline for the fine-tuning step. First, a textual encoding step transforms tabular data into meaningful text (a). Subsequently, a feature order permutation step is applied (b), before the obtained sentences can be used for the fine-tuning of a large language model LLM (c). The toy tabular data set inspired by the Adult Income data set (Dua & Graff, 2017).

et al., 2019; Brown et al., 2020) for natural language generation tasks showed the strength of the self-attention mechanism.

There are no prior works on the specific problem of generating realistic tabular with the help of pretrained LLMs. Our work is the first to explore this path, which leads to state-of-the-art performance.

3 GReaT: Generation of Realistic Tabular Data

This section presents the GReaT approach for fully-conditional tabular data generation using transformer-based neural networks. GReaT consists of two major stages: (1) the fine-tuning of a pretrained large language model (LLM) on a textually encoded tabular data set as shown in Fig. 2 and (2) sampling from the fine-tuned LLM to generate synthetic tabular data. We illustrate the sampling procedure in Fig. 3. In the following, we describe each component of the fine-tuning and sampling steps in detail. We conclude this section with a brief summary of our approach.

3.1 GReaT fine-tuning

Textual encoding. Standard pretrained generative LLMs expect sequences of words as inputs. Hence, to apply a LLM on tabular data, we have to convert each row of our data set into a textual representation. To this end, we propose the textual encoder.

Definition 1 (Textual encoder) *Given a tabular data set of n columns with feature names $f_1, f_2, \ldots, f_n$ and m rows of samples $\mathbf{s}_1, \ldots, \mathbf{s}_m$, we let the entry $v_{i,j}, i \in \{1, ..., n\}, j \in \{1, ..., m\}$ represent the value of the j-th feature of the i-th data point. Taking the feature name and value into account, each sample $\mathbf{s}_i$ of the table is transformed into a textual representation $\mathbf{t}_i$ using the following subject-predicate-object transformation:*

$$t_{i,j} = [f_j, \text{“}is\text{”}, v_{i,j}, \text{“},\text{”}] \qquad \forall i \in \{1, ..., n\}, j \in \{1, ..., m\}, \tag{1}$$

$$\mathbf{t}_i = [t_{i,1}, t_{i,2}, ..., t_{i,m}] \qquad \forall i \in \{1, ..., n\}, \tag{2}$$

where t_{ij}, the textually encoded feature, is a clause with information about a single value and its corresponding feature name, and $[\,\cdot\,]$ denotes the concatenation operator.

Remark 1. For the scope of this work, we treat the target variable as a regular feature in our formulations.

Fig. 2 (upper panels) shows an illustrative example of the proposed encoding technique. This result could be a sequence like *“Occupation is doctor, Gender is female, Age is 34,”*, which requires minimal preprocessing and does not suffer from information loss. While in standard natural language

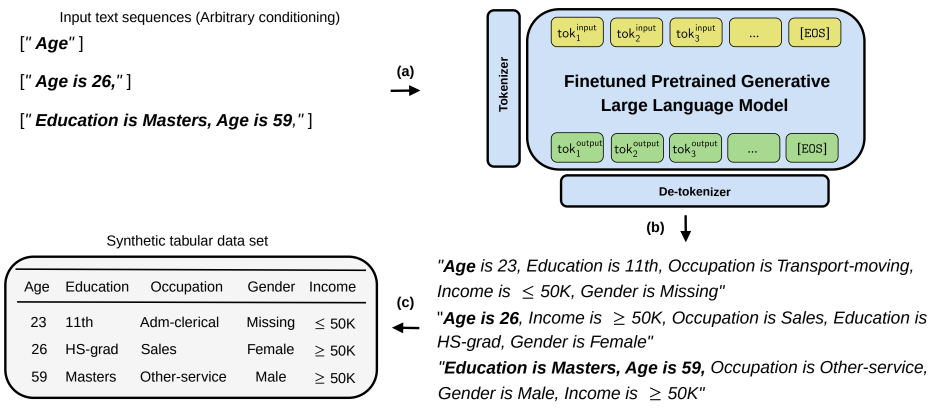

Input text sequences (Arbitrary conditioning)
[" Age"]
[" Age is 26,"]
[" Education is Masters, Age is 59,"]
(a)
Tokenizer
tok_1^input tok_2^input tok_3^input ... [EOS]
Finetuned Pretrained Generative Large Language Model
tok_1^output tok_2^output tok_3^output ... [EOS]
De-tokenizer
(b)
"Age is 23, Education is 11th, Occupation is Transport-moving, Income is ≤ 50K, Gender is Missing"
"Age is 26, Income is ≥ 50K, Occupation is Sales, Education is HS-grad, Gender is Female"
"Education is Masters, Age is 59, Occupation is Other-service, Gender is Male, Income is ≥ 50K"
(c)
Synthetic tabular data set
Age Education Occupation Gender Income
23 11th Adm-clerical Missing ≤ 50K
26 HS-grad Sales Female ≥ 50K
59 Masters Other-service Male ≥ 50K

Figure 3: The sampling procedure of the proposed method for synthetic data generation. In order to generate new data points using a pretrained LLM, it is necessary to transform either a single feature name or an arbitrary combination of feature-value pairs into text (a). Subsequently, the input is completed by the fine-tuned LLM (b) and can be transformed back into a tabular format (c). In comparison to other state-of-the-art approaches, GReaT allows arbitrary conditioning on feature subsets without model retraining, i.e., the sampling can be performed by conditioning on any feature name or combination of feature names and values.

sequences the word order is crucial, in our case there is no preferred order between the individual features.

Random feature order permutation. The transformation of a tabular feature vector into a sequence using the above textual subject-predicate-object encoding scheme artificially introduces pseudo-positional information into the tabular data sample, which is not natural for tabular data. Put simply, this means that there is no spacial ordering relationship between features in tabular data sets (Borisov et al., 2021). To reconstruct the feature order independence, we randomly permute the encoded short sentences $t_{i,j}$ of the full textual representation $\boldsymbol{t}_i$ by permutations $\boldsymbol{k}$.

Definition 2 (Feature order permutation function) *Formally, the result of applying a random feature order permutation* $\boldsymbol{k}$ *to* $\mathbf{t}_i$*, where each* $k_j \in \{1, ..., n\}$ *is arbitrary and* $k_j \neq k_{j'}$ *for* $j \neq j'$*, is defined as*

$$\mathbf{t}_i(\boldsymbol{k}) = [t_{i,k_1}, t_{i,k_2}, ..., t_{i,k_m}] \quad \forall i \in \{1, ..., n\}. \tag{3}$$

When using shuffled orders of the textually encoded features, we fine-tune our generative language model on samples *without order dependencies*. Moreover, using permutations of the above form is highly beneficial as they allow for *arbitrary conditioning* in tabular data generation, which is further discussed in the following subsection.

Fine-tuning a pretrained auto-regressive language model. Finally, we describe the fine-tuning procedure of a pretrained LLM to the encoded tabular data for generation tasks. We suppose a textually encoded tabular data set $\mathcal{T} = \{\mathbf{t}_i(\boldsymbol{k}_i)\}_{i=1,...,n}$ that was transformed into text by the proposed encoding scheme. Let $\boldsymbol{k}_i$ be randomly drawn permutations and n denote the number of rows.

To be processed with a LLM, the input sentences $\mathbf{t} \in \mathcal{T}$ need to be encoded into a sequence of tokens from a discrete and finite vocabulary $\mathcal{W}$. These tokens can be character, word or subword encodings such as the Byte-Pair-Encodings (Sennrich et al., 2015). Thus, $\boldsymbol{t} \in \mathcal{T}$ is represented by a sequence of tokens $(w_1, \ldots, w_j) = \text{TOKENIZE}(\mathbf{t})$ with tokens $w_1, \ldots, w_j \in \mathcal{W}$, where j denotes the number of tokens required to describe the character sequence $\mathbf{t}$. Commonly, the probability of natural-language sequences is factorized in an auto-regressive manner in LLMs (Jelinek, 1980; Bengio et al., 2000). It is represented as a product of output probabilities conditioned on previously observed tokens,

$$p(\boldsymbol{t}) = p(w_1, \ldots, w_j) = \prod_{k=1}^{j} p(w_k | w_1, ..., w_{k-1}). \tag{4}$$

In this formulation, it becomes evident that LLMs formally need to be highly capable predictors for follow-up tokens given an arbitrary-length sequence of preceding tokens. Indeed, the model is trained to output a probability distribution over possible next tokens w_k from an input sequence $w_1, ..., w_{k-1}$ of arbitrary length. The entire model is usually fitted by optimizing the parameters to maximize the probability $\prod_{\boldsymbol{t} \in \mathcal{T}} p(\boldsymbol{t})$ of the entire training data set.

As a result, an end-user can choose any existing *generative language* model for tabular data modeling and exploit the vast amount of contextual knowledge present in these models (Roberts et al., 2020). For instance, in generative transfomer-decoder LLM architectures (e.g., the GPT models (Radford et al., 2018; 2019; Brown et al., 2020)) word-embeddings are obtained from large corpus of text (e.g., GPT3 model trained on 45TB of textual data (Brown et al., 2020)). By learning from such large collection of data, Transformers are able to build robust contextualized representations of language (Liu et al., 2021). Fine-tuning enables the model to leverage this contextual information in combination with the feature and category names to boost the models capabilities in a manner that is similar to transfer learning by learning bidirectional representations (Raffel et al., 2020).

3.2 GReaT SAMPLING OF SYNTHETIC DATA

Having obtained a fine-tuned auto-regressive model $\boldsymbol{q}$ of the textual training data set that returns a categorical output distribution $\boldsymbol{z} = \boldsymbol{q}(w_1, \dots, w_{k-1})$ over possible follow up tokens for an input sequence $w_1, \dots, w_{k-1}$, we can apply several sampling strategies. Usually, the next token ω is sampled by weighted choice sampling with a temperature parameter $T > 0$ from the output $\boldsymbol{z}$ of the LLM,

$$p(\omega | w_1, \dots, w_{k-1}) = \frac{\boldsymbol{e}^{(\boldsymbol{z}_\omega / T)}}{\sum_{\omega' \in \mathcal{W}} \boldsymbol{e}^{(\boldsymbol{z}_{\omega'} / T)}}. \quad (5)$$

We note that the auto-regressive paradigm offers the possibility of sampling from token distributions $p(w_{k+1:j} | w_{1:k})$ with arbitrary preconditioning $w_{1:k}$. When we use random feature order permutations at train time, we can also start the textual sequence with any possible combination of features and values at inference time. As a result, the GReaT method is particularly flexible and could possibly be used in a variety of real-world problems such as missing value imputation (Kachuee et al., 2020) or generation of realistic counterfactual explanations (Pawelczyk et al., 2020). Moreover, the sampling of the conditional distribution comes at no additional costs.

Sampling and extraction of synthetic tabular data. Therefore, the setup provides several ways to sample new tabular data points using the GReaT method. We initialize the model with certain conditions and let the LLM sample the remaining tokens to complete the feature vector (in its textual representation). In total, we propose three options of preconditioning:

- **Feature Name Preconditioning.** Only a feature name, but no value is provided as an initial condition. This type of conditioning is able to generate samples from the entire joint data distribution $p(V_1, \dots, V_n)$, where V denotes the random variables representing the n features in the data set.
- **Name-Value Pair Preconditioning.** In this case, a single feature name and a value are provided. Starting from this input, GReaT will complete the sample. This approach will generate samples from the distribution $p(V_{\setminus\{i\}} | V_i = v_i)$. Because modeling a single feature distribution is usually tractable (by frequentist estimation for categorical features or by fitting a parametric density estimator for continuous features), we can first sample a value for V_i and then apply Name-Value Pair Preconditioning to sample the remaining features. Thereby, we can also sample the entire data distribution.
- **Multiple Name-Value Pair Preconditioning.** We can also provide multiple Name-Value pairs $V_{i_1}{=}v_{i_1}, \dots, V_{i_k}{=}v_{i_k}$ as precondition to the LLM to realize arbitrary conditioning. By providing the textual encoding of this condition, we are able to sample from the distribution of the remaining features effectively $p(V_{\setminus\{i_1,\dots,i_k\}} | V_{i_1}{=}v_{i_1}, \dots, V_{i_k}{=}v_{i_k})$.

We illustrate the GReaT sampling possibilities in Fig. 3 and Fig. 9 that underline the high flexibility of the proposed approach for synthetic data generation. We apply commonly accepted pattern-matching algorithms using regular expressions to convert the generated textual feature representations back to a tabular format (Aho, 1991). We dismiss the respective samples in the rare case where

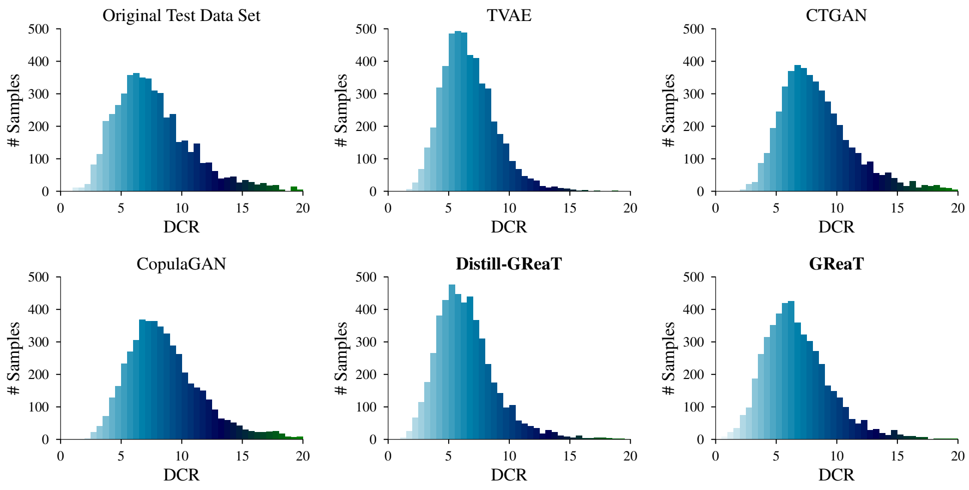


Figure 4: Distance to closest record (DCR) distributions for the California Housing data set with respect to the original train set. "Original Test Data Set" shows the DCR between the original test set and the original train set. This experiment shows that the proposed method does not "copy" samples from the training set but rather generates new synthetic samples close to the original samples.

the required format is violated. Generation rates of invalid samples were monitored and found to be consistently below 1 %.

A brief summary of the strengths of the GReaT method. GReaT comes with several significant advantages over related approaches: It (i) allows the end-user to have full probabilistic control over the sampling procedure by its arbitrary conditioning power; it (ii) utilizes the knowledge from large text data bases to obtain a better representation that includes context knowledge; (iii) the proposed approach is easy to use, since it does not require a preprocessing of the data values. There is no need to specify discrete or numerical variables beforehand and the information loss due to preprocessing is therefore kept at its bare minimum.

4 Experimental Evaluation

In this section, we empirically demonstrate the performance of the proposed GReaT approach using multiple qualitative and quantitative experiments. Lastly, for better reproducibility, we provide information on the packages and parameters for the selected LLMs.

Data sets. For the evaluation of the proposed algorithm, we utilize four real-world data sets that come from various domains. They also come in different sizes, reaching from less than 1,000 to over 32,000 samples. We also consider three synthetic data sets with a varying numbers of features. Key characteristics of each data set are presented in Table 8. We split all data sets into train and test sets to avoid any data leakage. All models are trained or fine-tuned on the same training data samples. To demonstrate the power of our synthetic data generation framework to work out-of-the-box with real-world data, we apply zero data preprocessing, e.g., feature names and values are used as they are presented in original data sets.

Baselines. As baselines for our experiments, we use three deep learning-based methods. CTGAN (Xu et al., 2019) is based on a generative adversarial network (GAN) (Goodfellow et al., 2014) for tabular data that allows to condition the generation process on a only single discrete feature. The same authors proposed TVAE (Xu et al., 2019), an variational autoencoder (VAE) for tabular data. The CopulaGAN model from the Synthetic Data Vault (SDV) framework (Patki et al., 2016) is applying the Gaussian copulas to simplify the underlying CTGAN.

We compare those baselines to the proposed method with two different pretrained transformer-decoder LLM models of various sizes. The smaller, distilled version of GPT-2 (Sanh et al., 2019) has 82 million learned parameters. We term the corresponding tabular data generator Distill-GReaT. An original version of GPT-2 by Radford et al. (2019) has over 355 million trainable parameters – we refer to this version simply as GReaT. A description of the architecture of the selected generative LLMs can be found in Table 9. We apply name-value pair preconditioning to start sampling.

For the evaluation of synthetic tabular data, we select four measures, all of which have been featured in multiple previous studies on synthetic tabular data generation (Borisov et al., 2021).

Machine learning efficiency (MLE). Since the generated data set should be able to replace the real data in a training process, this measure evaluates the performance of discriminative models trained on synthetic data sets. Therefore, the models are tested on real test data, and the scores are compared to the original performance when the models were trained on the original, real training data set. Table 1 and Table 4 present the results for the machine learning efficiency of proposed generative models compared to the baseline models. To be independent of the concrete discriminative model, we evaluated the synthetic data sets using a Linear/Logistic Regression (LR), a Decision Tree (DT) and a Random Forest (RF) (Ho, 1995). We observe that either GReaT and Distill-GReaT outperform all competitors and entail considerable performance improvements.

		Original	TVAE	CopulaGAN	CTGAN	**Distill-GReaT**	**GReaT**
TR (↑)	LR	82.72±0.00	79.58±0.00	73.30±0.00	73.30±0.00	78.53±0.00	**80.10±0.00**
	DT	89.01±0.00	81.68±1.28	73.61±0.26	73.30±0.00	77.38±0.51	**83.56±0.42**
	RF	85.03±0.53	81.68±1.19	73.30±0.00	71.41±0.53	79.90±0.53	**84.30±0.33**
HE (↑)	LR	71.80±0.00	71.04±0.00	42.03±0.00	57.72±0.00	70.58±0.00	**71.90±0.00**
	DT	81.90±0.02	76.39±0.10	42.36±0.10	61.34±0.09	**81.40±0.15**	79.10±0.07
	RF	83.19±0.21	77.24±0.25	42.35±0.34	62.35±0.35	**82.14±0.13**	80.93±0.28
AD (↑)	LR	85.00±0.00	80.53±0.00	80.61±0.00	83.20±0.00	84.65±0.00	**84.77±0.00**
	DT	85.27±0.01	82.80±0.08	76.29±0.06	81.32±0.02	84.49±0.04	**84.81±0.04**
	RF	85.93±0.11	83.48±0.11	80.46±0.21	83.53±0.07	85.25±0.07	**85.42±0.05**
CH (↓)	LR	0.40±0.00	0.65±0.00	0.98±0.00	0.61±0.00	0.57±0.00	**0.34±0.00**
	DT	0.32±0.01	0.45±0.01	1.19±0.01	0.82±0.01	0.43±0.01	**0.39±0.01**
	RF	0.21±0.01	0.35±0.01	0.99±0.01	0.62±0.01	0.32±0.01	**0.28±0.01**

Table 1: ML efficiency experiment. The best results are marked in **bold**, the second-best results are underlined. Four real-world data sets are used, Travel Customers (TR), HELOC (HE), Adult Income (AD), and California Housing (CH). Each data set is evaluated on three discriminative ML models, Linear/Logistic Regression (LR), Decision Tree (DT), and Random Forest (RF). For classification tasks the accuracy score is reported, in case of the regression the mean squared error is used. Results are averages over five trials with different random seeds (cf. Appendix A for additional measures).

Distance to closest records (DCR) histogram. To verify that the generated data is similar to original samples while not being exact copies, this measure computes the distance to the closest record in original training data set $\mathcal{T}_{train}$. For each synthetic record s_{gen}, it is given by $\text{DCR}(s_{gen}) = \min\{\text{Distance}(s_{gen}, s_i) | s_i \in \mathcal{T}_{train}\}$. As a distance measure, we use the L_1-norm of the differences. We set the difference to be 0 for equal for categorical features and to 1 otherwise. In the best case, all DCR scores are non-zero and their distribution is close to that of DCRs computed with points from the original test data set $\mathcal{T}_{test}$. Visualizations of the distribution of the minimal distances can be found in Fig. 4 and Appendix A. Indeed, the results show that our model generates unseen samples in the expected proximity to the original ones. The same holds for all used models and we did not observe large differences in this metric overall.

Discriminator measure. To check whether the generated data cannot be easily told apart from the original data, we train a Random Forest discriminator (with hyperparameter tuning) on a mix of the generated train set (with label 0) and the original train set (with label 1). We then report the test accuracy on a test data set, which contains equal shares of samples from the generated test set and the real test set. Scores are shown in Table 2 and demonstrate superior performance of the GReaT approach, which on average (across all used data sets) decreases the discriminator performance by 16.2 % over other competitive baselines for tabular data generation.

Bivariate joint distribution plots. We qualitatively compare the generated feature distributions by the baselines and the GReaT approach to that of the original data. As an illustrative example, we present joint density plots for the `Longitude` and `Latitude` features in the Caifornia Housing data set in Fig. 1. While CTGAN and CopulaGAN fail to model the strong dependency between

Data set	CopulaGAN	TVAE	CTGAN	**Distill-GReaT**	**GReaT**
Adult	88.54±0.09	88.49±0.18	97.23±0.10	69.79±0.17	**62.84±0.08**
HELOC	97.83±0.12	100.00±0.00	99.99±0.01	**68.30±0.47**	69.15±0.36
California	85.48±0.16	85.04±0.27	82.98±0.27	76.18±0.15	**70.68±0.30**
Travel	78.19±0.33	72.80±0.50	**71.26±0.40**	85.84±0.43	78.68±0.26
Average	87.51±0.04	86.58±0.06	87.86±0.05	75.03±0.08	**70.34±0.06**

Table 2: Discriminator measure (accuracy in %). Lower accuracy values indicate that the discriminator cannot differentiate between fake records and real samples. The accuracy of a perfectly indistinguishable data set would be 50%. Best results are **bold**, second-best results are underlined.

	Adult		HELOC		Calfornia		Travel	
Distill-GReaT	discr. (↓)	MLE (↑)	discr.(↓)	MLE (↑)	discr. (↓)	MLE (↓)	discr.(↓)	MLE (↑)
w/o permutation	**61.18±0.16**	**85.71±0.07**	79.18±0.23	81.97±0.55	76.47±0.26	**0.26±0.01**	**61.62±0.67**	**83.77±0.33**
with permutation	69.77±0.09	85.25±0.07	**68.29±0.34**	**82.14±0.13**	**76.09±0.14**	0.33±0.01	85.65±0.38	80.31±0.53
w/o pretraining	99.14±0.02	84.15±0.05	99.51±0.01	76.32±0.20	85.10±0.22	0.34±0.01	**62.98±0.71**	**81.47±0.42**
with pretraining	**69.77±0.09**	**85.25±0.07**	**68.29±0.34**	**82.14±0.13**	**76.09±0.14**	**0.33±0.01**	85.65±0.38	80.31±0.53

Table 3: Results of experiments with and without permutation, as well as with and without pretraining based on discrimination (discr.) and ML efficiency (MLE, measured by the accuracy of a Random Forest model) on four real-world data sets. In all experiments, we used Distill-GReaT with the same training and pretraining setup; the only difference is the input data and pretraining.

these variables (indicated by out-of-distribution samples having both high latitude and longitude or low values for both), Distill-GReaT and GReaT yield densities that align well with the ground truth boundaries. TVAE shows mediocre performance but still places density mass outside the bounds.

Effects of pretraining and permutations. Having obtained impressive results with the complete setup, we investigate the role of the individual components towards its success. We compare the full model (with permutation and pretraining step) to a model without permutation and a model without pretraining. Tab. 3 presents ML efficiency and discriminator results for the modified models. On all but the very small "Travel" data set (954 samples in total), we observe pretraining to help boost the performance. This might be due to the context knowledge made available to GReaT through the extensive adaptation to large text corpora. The results regarding the feature order permutation step are mixed – the MLE performance decreases with permutations but the results for the discriminator metric are inconclusive. With permutations, the learning problem is undoubtedly more demanding. It provides models with the ability to perform generation based on arbitrary conditioning. However, we conjecture that it might also increase performance in some cases because it does not introduce any possibly unnatural, fixed feature ordering into the tabular data.

Reproducibility details. We utilize pretrained generative language models from the established HuggingFace framework (Wolf et al., 2020). Its routines are also used for the fine-tuning and sampling steps. We plan to open-source all code online. Further details about all experiments, such as hyperparameters for each baselines and experiment setups are provided in the Appendix B.

```
# pip install be-great
from great import GReaT
model = GReaT('gpt2')
model.fit(dataset, epochs=100)
synthetic_samples = model.sample(n_samples=1000)
```

Figure 5: Python pseudo-code to import our model after it was pip-installed.

5 Discussion

Processing of numerical values with LLMs in the GReaT approach. Since we convert heterogeneous tabular data into text (Sec. 3), continuous and discrete numerical values are represented as character sequences. While this might seem counter-intuitive at first, multiple independent studies have shown that transformer-based models are capable of understanding and processing numerical data encoded in such a way (Wallace et al., 2019; Brown et al., 2020). This is even true in the one-shot setting for inference tasks (Dinh et al., 2022). The impressive performance of GReaT that encompasses the numerical features aligns with these previous observations. Having said that, smarter encodings for numerical values can also be considered a possible path to further improvement.

Unification of multiple modalities through transformer models. Along with the usual numerical values, tabular data frequently contains textual metadata, e.g., feature names, named categories ("male", "female"), and open text features (e.g., `remarks`). The recent evolution of transformer neural networks now permits to holistically unite this information, which was processed separately in the past, and learn context-specific, robust, and meaningful representations. In the case of tabular data, information comes from the textual and numerical data modalities. The GReaT approach and the results presented in this work provide an initial clue of the range of possibilities and opportunities that may lie in this line of research.

6 Conclusion

In our recent work, we investigate how state-of-the-art generative language models can be leveraged to synthesize highly realistic tabular data samples. Instead of following the usual path of encoding heterogeneous tabular data in a numerical format, we devise a textual encoding strategy and represent it with sentences that capture each record's semantics. The resulting transformer-decoder network fine-tuned on this data exhibits unprecedented generative performance and outstanding flexibility at the same time. We term our method **G**eneration of **Rea**listic **T**abular data (**GReaT**). GReaT unites several remarkable characteristics that address key problems in tabular data modeling: First, minimal preprocessing is required, which is efficient and results in the least possible information loss, thereby tackling the issue of possibly *lossy preprocessing*. Second, leveraging random feature order permutations, we exploit the capability of *arbitrary conditioning* and are thus equipped with full control over the probabilistic sampling procedure for tabular data. Finally, through pretraining, we can incorporate *contextual and semantic knowledge* extracted from terabytes of textual data for a more authentic tabular data synthesis. Based on all empirical evidence presented in our work, we could demonstrate that heterogeneous tabular data – if transformed coherently into sentences that capture the semantics of feature names and values – can be modeled and generated by pretrained large language models for generative tasks. In conclusion, we see our work as a door opener leading to yet undiscovered possibilities in the domain of heterogeneous data generation.

References

Alfred V Aho. Algorithms for finding patterns in strings, handbook of theoretical computer science (vol. a): algorithms and complexity, 1991.

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. A neural probabilistic language model. *Advances in neural information processing systems (NeurIPS)*, 13, 2000.

Omar Benjelloun, Shiyu Chen, and Natasha Noy. Google dataset search by the numbers. In *International Semantic Web Conference*, pp. 667–682. Springer, 2020.

Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. Deep neural networks and tabular data: A survey. *arXiv preprint arXiv:2110.01889*, 2021.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems (NeurIPS)*, 33:1877–1901, 2020.

Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pp. 108–122, 2013.

Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. *Advances in neural information processing systems (NeurIPS)*, 32, 2019.

Edward Choi, Siddharth Biswal, Bradley Malin, Jon Duke, Walter F. Stewart, and Jimeng Sun. Generating Multi-label Discrete Patient Records using Generative Adversarial Networks. *Machine learning for healthcare conference*, pp. 286–305, 2017.

CKCN Chow and Cong Liu. Approximating discrete probability distributions with dependence trees. *IEEE transactions on Information Theory*, 14(3):462–467, 1968.

Sajad Darabi and Yotam Elor. Synthesising multi-modal minority samples for tabular data. *arXiv preprint arXiv:2105.08204*, 2021.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Tuan Dinh, Yuchen Zeng, Ruisu Zhang, Ziqian Lin, Shashank Rajput, Michael Gira, Jy-yong Sohn, Dimitris Papailiopoulos, and Kangwook Lee. Lift: Language-interfaced fine-tuning for non-language machine learning tasks. *arXiv preprint arXiv:2206.06565*, 2022.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Adrià Gascón, Phillipp Schoppmann, Borja Balle, Mariana Raykova, Jack Doerner, Samee Zahur, and David Evans. Privacy-preserving distributed linear regression on high-dimensional data. *Proc. Priv. Enhancing Technol.*, 2017(4):345–364, 2017.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pp. 278–282. IEEE, 1995.

Frederick Jelinek. Interpolated estimation of markov source parameters from sparse data. In *Proc. Workshop on Pattern Recognition in Practice, 1980*, 1980.

Mohammad Kachuee, Kimmo Karkkainen, Orpaz Goldstein, Sajad Darabi, and Majid Sarrafzadeh. Generative imputation and stochastic prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

Sanket Kamthe, Samuel Assefa, and Marc Deisenroth. Copula flows for synthetic data generation. *arXiv preprint arXiv:2101.00598*, 2021.

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8110–8119, 2020.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Aki Koivu, Mikko Sairanen, Antti Airola, and Tapio Pahikkala. Synthetic minority oversampling of vital statistics data with generative adversarial networks. *Journal of the American Medical Informatics Association*, 27(11):1667–1674, 2020.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7871–7880, 2020.

Wei-Chao Lin and Chih-Fong Tsai. Missing value imputation: a review and analysis of the literature (2006–2017). *Artificial Intelligence Review*, 53(2):1487–1509, 2020.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*, 2021.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Chao Ma, Sebastian Tschiatschek, Richard Turner, José Miguel Hernández-Lobato, and Cheng Zhang. Vaem: a deep generative model for heterogeneous mixed type data. *Advances in Neural Information Processing Systems (NeurIPS)*, 33, 2020.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

Alejandro Mottini, Alix Lheritier, and Rodrigo Acuna-Agost. Airline Passenger Name Record Generation using Generative Adversarial Networks. *arXiv preprint arXiv:1807.06657*, 2018.

R Kelley Pace and Ronald Barry. Sparse spatial autoregressions. *Statistics & Probability Letters*, 33 (3):291–297, 1997.

Inkit Padhi, Yair Schiff, Igor Melnyk, Mattia Rigotti, Youssef Mroueh, Pierre Dognin, Jerret Ross, Ravi Nair, and Erik Altman. Tabular transformers for modeling multivariate time series. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3565–3569. IEEE, 2021.

Noseong Park, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. Data synthesis based on generative adversarial networks. *Proceedings of the VLDB Endowment*, 11(10):1071–1083, 2018.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

N. Patki, R. Wedge, and K. Veeramachaneni. The synthetic data vault. In *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 399–410, Oct 2016. doi: 10.1109/DSAA.2016.49.

Martin Pawelczyk, Klaus Broelemann, and Gjergji Kasneci. Learning model-agnostic counterfactual explanations for tabular data. In *Proceedings of The Web Conference 2020*, pp. 3126–3132, 2020.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

Adam Roberts, Colin Raffel, and Noam Shazeer. How much knowledge can you pack into the parameters of a language model? *arXiv preprint arXiv:2002.08910*, 2020.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. In *NeurIPS EMC^2 Workshop*, 2019.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*, 2015.

Sandeep Subramanian, Sai Rajeswar, Francis Dutil, Christopher Pal, and Aaron Courville. Adversarial generation of natural language. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pp. 241–251, 2017.

L Vivek Harsha Vardhan and Stanley Kok. Generating privacy-preserving synthetic tabular data using oblivious variational autoencoders. In *Proceedings of the Workshop on Economics of Privacy and Data Labor at the 37 th International Conference on Machine Learning (ICML)*, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems (NeurIPS)*, 30, 2017.

Eric Wallace, Yizhong Wang, Sujian Li, Sameer Singh, and Matt Gardner. Do NLP models know numbers? probing numeracy in embeddings. In *Empirical Methods in Natural Language Processing*, 2019.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pp. 38–45, 2020.

Lei Xu, Maria Skoularidou, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Modeling tabular data using conditional GAN. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, 2019.

A Additional Experimental Results

A.1 Further MLE measures

Next to the accuracy measures (Tab. 1) we also report the ROCAUC score and the F1 score for the ML efficiency experiment (Sec. 4).

		Original		TVAE		CopulaGAN		CTGAN		**Distill-GReaT**		**GReaT**	
		ROCAUC	F1	ROCAUC	F1	ROCAUC	F1	ROCAUC	F1	ROCAUC	F1	ROCAUC	F1
TR (↑)	LR	81.90±0.00	81.84±0.00	80.14±0.00	78.73±0.00	60.78±0.00	62.00±0.00	77.70±0.00	62.00±0.00	79.61±0.00	77.44±0.00	**81.47**±**0.00**	**79.53**±**0.00**
	DT	95.74±0.00	88.34±0.00	**83.32**±**1.93**	81.78±1.20	50.59±0.48	62.73±0.60	56.89±0.00	62.00±0.00	71.50±0.62	77.01±0.45	80.35±0.13	**83.63**±**0.38**
	RF	94.15±0.62	84.68±0.49	87.25±0.45	80.45±1.18	51.56±0.31	62.00±0.00	66.64±1.12	65.37±1.35	87.90±0.32	79.79±0.48	**89.90**±**0.30**	**84.33**±**0.37**
HE (↑)	LR	79.43±0.00	71.79±0.00	77.05±0.00	71.01±0.00	43.04±0.00	41.90±0.00	62.73±0.00	57.63±0.00	77.44±0.00	70.59±0.00	**78.88**±**0.00**	**71.47**±**0.00**
	DT	89.52±0.04	81.81±0.03	82.56±0.18	76.39±0.10	35.98±0.20	42.12±0.11	62.18±0.12	61.24±0.09	**89.10**±**0.11**	**81.40**±**0.15**	88.80±0.13	78.68±0.07
	RF	90.52±0.13	83.15±0.20	85.29±0.20	77.20±0.25	38.60±0.35	42.27±0.33	65.34±0.10	62.29±0.35	**89.81**±**0.10**	**82.12**±**0.13**	89.07±0.09	80.71±0.29
AD (↑)	LR	90.48±0.00	84.55±0.00	87.15±0.00	81.38±0.00	81.92±0.00	79.80±0.00	87.86±0.00	83.19±0.00	89.52±0.00	**84.60**±**0.00**	**90.25**±**0.00**	84.52±0.00
	DT	89.60±0.05	84.31±0.01	84.68±0.14	82.36±0.07	73.41±0.06	76.22±0.04	84.47±0.06	80.76±0.02	**88.20**±**0.12**	83.57±0.05	88.07±0.12	**84.29**±**0.04**
	RF	91.45±0.04	85.21±0.10	88.73±0.07	83.44±0.09	77.53±0.19	79.11±0.17	88.47±0.06	83.00±0.08	90.54±0.03	84.57±0.10	**90.77**±**0.06**	**84.85**±**0.04**

Table 4: Additional results of the ML efficiency experiments measuring the ROCAUC score and the F1 score for the three real-world classification data sets, Travel Customers (TR), HELOC (HE), and Adult Income (AD). Each data set is evaluated on three discriminative ML models, Linear/Logistic Regression (LR), Decision Tree (DT), and Random Forest (RF). The best results are marked in **bold**, the second-best results are underlined. Results are averages over five trials with different random seeds.

A.2 Average Negative Log-Likelihood Metric for synthetic data

The generated data should be likely under the training data's distribution. Therefore, we generate three common synthetic data sets, and compute both the likelihood of the sampled data under the training data's density ($\mathcal{L}_{syn}$). This can however be prone to overfitting, which is why we additionally deploy the likelihood fitness metric proposed by Xu et al. (2019) ($\mathcal{L}_{test}$). To compute this metric, a parametric density model (BNs and GMMs respectively) is fitted to the *generated* data. Then the likelihood of the original test samples is computed. The results in Table 5 indicate that LLMs are comparable with state-of-the-art deep neural networks when modeling high-dimensional mixture distributions.

	GMM		Asia (BN)		Alarm (BN)	
Model	$\mathcal{L}_{syn}$	$\mathcal{L}_{test}$	$\mathcal{L}_{syn}$	$\mathcal{L}_{test}$	$\mathcal{L}_{syn}$	$\mathcal{L}_{test}$
Identity	-4.403	-4.403	-2.265	-2.265	-11.739	-11.739
CopulaGAN	-4.406	**-4.676**	-6.321	-3.129	-22.438	-16.828
TVAE	**-4.205**	-4.817	-2.418	**-2.289**	-11.919	**-12.074**
CTGAN	-4.427	-4.723	-3.999	-2.528	-20.804	-15.248
Distill-GReaT	-4.372	-5.124	**-1.925**	-2.355	**-6.941**	-12.689

Table 5: Average Log-likelihood of synthetic data sampels on a density model derived from the original data ($\mathcal{L}_{syn}$) and of the original test data on the model derived from the syntethic data ($\mathcal{L}_{test}$). The best results are marked in **bold**, the second-best results are underlined.

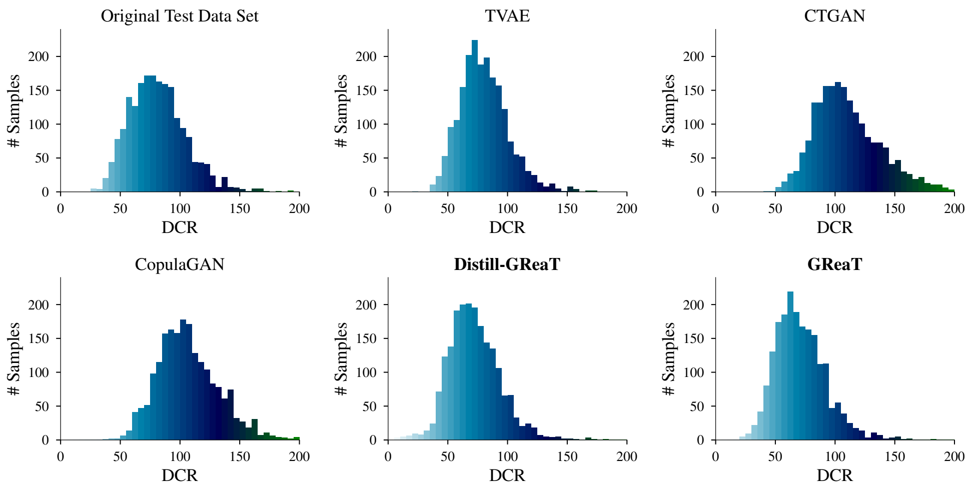


Figure 6: Distance to closest record (DCR) distributions for the HELOC Data set with respect to the original train set. "Original Test Data Set" shows the DCR between the original test set and the original train set. According to this experiment, the proposed method does not "copy" samples from the training set but rather generates new synthetic samples close to the original samples.

A.3 Distance to Closest Record Results

We compare the distribution of the minimal distances of the generated samples to the training data set. Figure 4 shows the distribution for the California Housing data set and Figure 6 for the HELOC data set. The results indicate that the generated samples are close to the original ones without coping them exactly.

A.4 Additional Qualitative Analysis Results

In Figure 7 we additionally compared the joint feature distribution of two exemplary features from the Adult Income data set, `Age` and `EducationNum`. The joint plots were computed using a kernel density estimator.

Travel Customers	https://www.kaggle.com/datasets/tejashvi14/tour-travels-customer-churn-prediction
HELOC	https://www.kaggle.com/datasets/averkiyoliabev/home-equity-line-of-creditheloc
Adult Income	https://archive.ics.uci.edu/ml/datasets/Adult/
California Housing	https://www.kaggle.com/datasets/camnugent/california-housing-prices

Table 6: URLs for real-world data sets of the study.

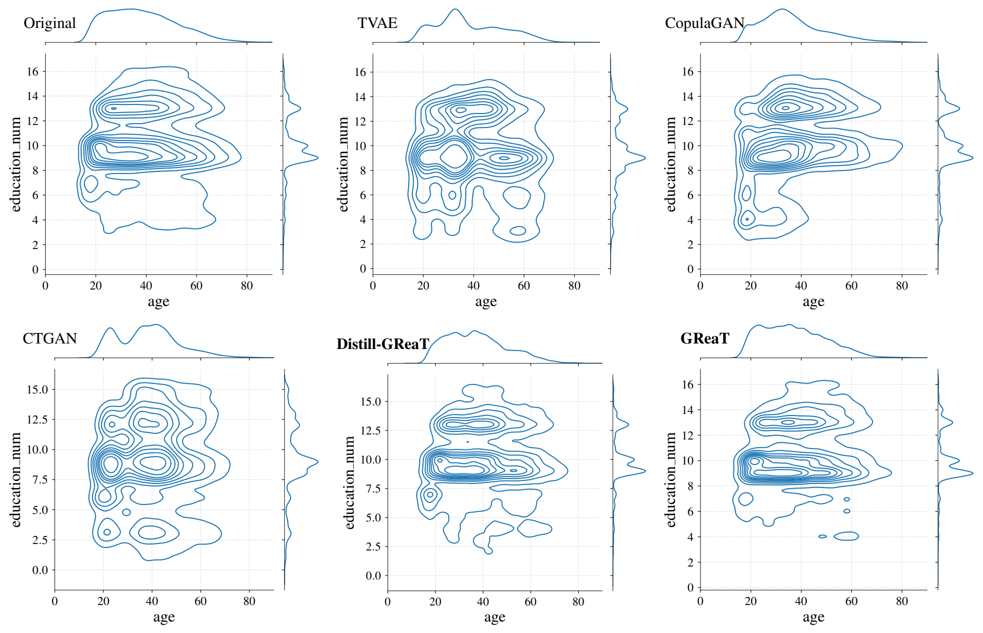


Figure 7: A comparison of the original and generated data sets for the Adult Income data set using joint plots for two related features – `Age` and `EducationNum`.

A.5 Runtime Comparison

We also analyse the training/fine-tuning and sampling time between baseline models and two versions of the GReaT method. Table 7 summarizes our time-benchmarking results. To make a fair comparison, we used the latest available versions of the corresponding implementations. Also, we utilize the same DL framework - PyTorch (Paszke et al., 2019), and the same number of epochs 200 as well as the sampling size - 1000, for each model.

	TVAE	CopulaGAN	CTGAN	Distill-GReaT	GReaT
training / fine-tuning time	0:46 min	2:30 min	1:10 min	1:35 h	9:10 h
sampling time	0.28 sec	0.119 sec	0.045 sec	4 sec	17 sec

Table 7: A run time comparison of all generative models of our study. Selected models were trained/fine-tuned for 100 epochs and 1000 samples were generated.

	Domain	#Samples	#Num	#Cat	Task	#Classes
(TR) Travel Customers	Churn	954	2	4	Classification	2
(HE) HELOC	Financial	9,871	21	2	Classification	2
(AD) Adult Income	Social	32,561	6	8	Classification	2
(CH) California Housing	Real Estate	20,640	8	0	Regression	-
Alarm	Synthetic	20,000	0	37	-	-
Asia	Synthetic	20,000	0	8	-	-
GMM	Synthetic	6,000	2	0	-	-

Table 8: Details of the real-world and synthetic data sets used in the experimental evaluations. #Num and #Cat columns indicate numbers of numerical and categorical features in each data set.

	#Parameters	#Layers	#Heads	Embedding Size	Context Size
Distill GPT-2	82M	6	12	768	1024
GPT-2	355M	24	16	1024	1024

Table 9: Structural details about the pretrained large language models used in our study.

	Single-Variable Conditional Sampling	Multi-Variable Conditional Sampling	Usage of the Context (Variable Names)	Transfer Learning	Data Prepossessing
CopulaGAN	✓	✓	✗	✗	Scaling, Encoding
TVAE	✗	✗	✗	✗	Scaling, Encoding
CTGAN	✓	✗	✗	✗	Scaling, Encoding
GReaT	✓	✓	✓	✓	Format conversion

Table 10: An analysis of the main properties of the synthetic generation frameworks for tabular data.

B Reproducibility Details

We fine-tune the Distill-GReaT model for each data set for 200 epochs, except for the California housing data set, for it, we fine-tune it for 100 epochs. The GReaT baseline is fine-tuned for 110, 310, 400, 255 epochs for California, Adult, Travel, and HELOC data sets, respectively. Depending on the GPU memory limitations, we vary the batch size from 8 to 124. For the sampling step, we set the temperature parameters T to 0.7 for all experiments and data sets. We sample new synthetic data using the name-value pair preconditioning (Sec. 3), starting with the target feature for all data sets (see an example in the supplementary materials).

	LR	DT	RF	
	max_iter	max_depth	max_depth	n_estimators
Travel Customers	100	6	12	75
HELOC	500	6	12	78
Adult Income	1000	8	12	85
California Housing	-	10	12	85

Figure 8: The hyperparameter configuration of the evaluation models for the ML efficiency experiments.

We utilize the AdamW optimizer (Loshchilov & Hutter, 2017) for the proposed generative models, with the learning rate 5×10^{-5}. We plan to share trained weights for the proposed models. The baseline models are trained for 200 epochs for each data set.

Our hardware setup consisted of two NVIDIA 2080RTX GPU with 12 GB RAM each, 126 GB system RAM, and AMD Ryzen 3960X with 24 cores, we use the Ubuntu 20.04 operation system.

For the ML efficiency and discriminator experiments (Sec, 4) we additionally use linear/logistic regression, decision tree, and random forest models from the Scikit-Learn package (Buitinck et al., 2013), we report the exact hyperparameters for each model in Table 8. For the discriminator measure experiment (Table 2), we tune hyperparameters for each data set using the 5-fold cross-validation.

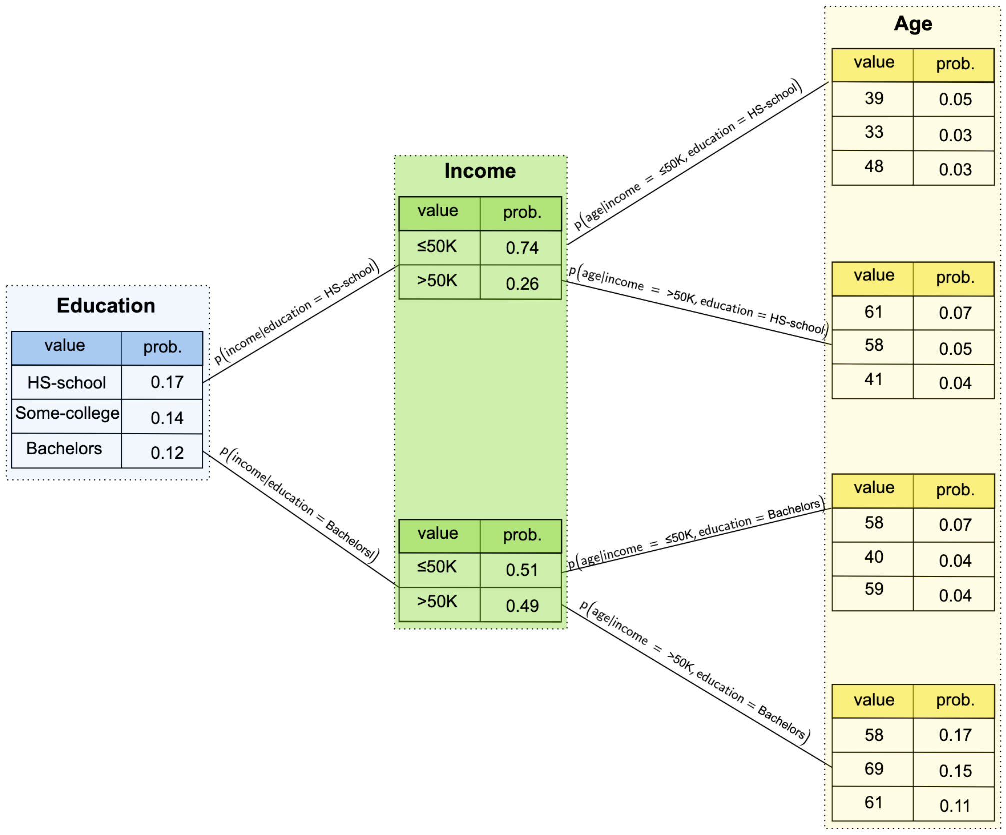


Figure 9: Example of the arbitrary conditioning using the GReaT approach on Adult data set. For this experiment we select only three variables `Education`, `Income`, and `Age` from the Adult Income data set (Dua & Graff, 2017). However, the proposed method can be scaled to the arbitrary number of conditions. The results obtaining by changing the input textual sequence, e.g., `Education is HS-school, Income is >50K, Age is`, after we obtain the conditional discriminate distribution $p(Age|income => 50K, education = HS - school)$. Interestingly, the GReaT method successfully learned that there is only two options for the `Income` variable. The arbitrary sampling is supported by our Python programming framework.

A.2 Explainable Deep Learning

A.2.1 CancelOut: A Layer for Feature Selection in Deep Neural Network

Publication: Published in the 28th International Conference on Artificial Neural Networks (ICANN), 2019.

Contribution: I conceived the idea for a new type of layer for deep neural networks and collaborated with Johannes Haug to conduct experiments assessing its effectiveness. I developed the code for this layer in all major deep learning frameworks and received invaluable feedback from Gjergji Kasneci regarding the structure of the paper. All co-authors, myself included, contributed to the revision process of the final manuscript.

CancelOut: A layer for feature selection in deep neural networks

Vadim Borisov[1]*, Johannes Haug[1], and Gjergji Kasneci[1,2]

[1] Eberhard Karls University of Tübingen, Tübingen, Germany
[2] SCHUFA Holding AG, Wiesbaden, Germany
`{vadim.borisov,johannes-christian.haug,gjergji.kasneci}@uni-tuebingen.de`

Abstract. Feature ranking (FR) and feature selection (FS) are crucial steps in data preprocessing; they can be used to avoid the curse of dimensionality problem, reduce training time, and enhance the performance of a machine learning model. In this paper, we propose a new layer for deep neural networks - CancelOut, which can be utilized for FR and FS tasks, for supervised and unsupervised learning. Empirical results show that the proposed method can find feature subsets that are superior to traditional feature analysis techniques. Furthermore, the layer is easy to use and requires adding only a few additional lines of code to a deep learning training loop. We implemented the proposed method using the PyTorch framework and published it online [3].

Keywords: Deep Learning · Feature Ranking · Feature Selection · Unsupervised Feature Selection · Machine Learning Explainability

1 Introduction

Feature importance and interpretability of machine learning (ML) models have received much attention in the recent years due to the fact that accurate estimations are not always enough to solve a data problem. An explanation of machine learning model outcomes may help not only to understand the model's results, but also to introduce new tests, better understand the data, and as a consequence from above, improve trust in the model, which is important when the model is used by specialists from other fields. However, most accurate and robust ML models usually cannot be interpreted [4].

One of the most effective ML methods nowadays is deep learning (DL), which can be explained in terms of the universal approximation theorem [2]. Which principally states that any compactly supported continuous function on $\mathbb{R}^n$ can be approximated with a single hidden layer feed-forward neural network (NN). However, due to DL's high inherent complexity, most DL models are primarily handled as a black box. Even though recent attempts have been made to address the issue of their interpretability and feature selection [4,1], existing methods are complicated.

[3] The code is available at: `www.github.com/unnir/CancelOut`

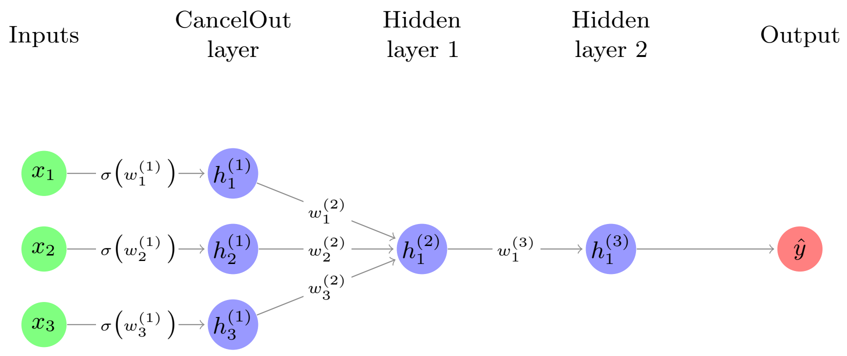


Fig. 1: A deep neural network with a CancelOut layer as an input layer.

2 Related work

Many research articles on feature ranking (FR) and feature selection (FS) using DL propose the permutation method, which is based on the idea that if we remove or corrupt a feature from a dataset, the performance will change. By analyzing these changes, it is possible to determine if a feature is valuable or not. The obvious drawback of this approach is that it is computation intensive, and in order to check n features, one needs to train a DL model at least n times.

Another similar strategy was proposed in [1], where the dropout layer is exploited for feature ranking [12]. To analyze which features are important, an artificial NN model with a dropout layer must achieve minimum loss, and the dropout layer should learn low dropout rates for important features while increasing the dropout rate for the rest of the unimportant features. In this case, a model can be run once.

An interesting approach for quantifying the influence of individual input signals on the output computed by a deep neural network was proposed in the paper [6]. This method is based on the estimation of local linear models for each neuron in the network and the propagation and aggregation of these models into a net wide model.

The work in [9] introduces a similar idea to the linear models with elastic net regularization, but it employs a NN with multiple layers. This method regularizes input weights in a loss function using $l1$ and $l2$ norms together; without these terms the method is unstable.

Furthermore, many articles investigate an interpretation of a decision of the ANN for a single data sample [13]. However, this approach is hard to apply for feature ranking of a whole dataset.

The paper is divided into four sections. In Section 3, the proposed approach for feature ranking is introduced. Section 4 presents the implementation and result of the study. Finally, Section 5 contains a summary of the work.

3 CancelOut

In this Section, we present a new layer for deep neural networks - CancelOut, a method that helps identify a subset of relevant input features (variables) in a dataset. Also, the proposed method can be applied for feature sensitivity analysis.

CancelOut is an artificial neural network (ANN) layer, which is comparable to a fully connected (FC) layer with one distinction: neurons in the FC layer have connections to every input, whereas neurons in the CancelOut layer have only one connection to one particular input (Fig. 1).

The primary idea behind CancelOut layer is to update its weights (W_{CO}) during the training stage so that irrelevant features will be *canceled out* with a negative weight (Eq. 1). Otherwise, the best variables, which contribute more to a learning process, are going to be passed through with a positive weight.

$$CancelOut(\boldsymbol{X}) = \boldsymbol{X} \odot g(W_{CO}) \tag{1}$$

where $\odot$ indicates an element-wise multiplication, $\boldsymbol{X}$ is an input vector $\boldsymbol{X} \in \mathbb{R}_v^N$, W_{CO} is a weight vector $W_{CO} \in \mathbb{R}_v^N$, N_v is the feature size, and g is an activation function. Note, $g(x)$ denotes here element-wise application, e.g. $\boldsymbol{X} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$, then $g(\boldsymbol{X}) = g\left(\begin{bmatrix} a \\ b \\ c \end{bmatrix}\right) = \left(\begin{bmatrix} g(a) \\ g(b) \\ g(c) \end{bmatrix}\right)$.

3.1 Theoretical Justification of the CancelOut layer

For simplicity, consider a three layers artificial neural network (Fig. 1) with *linear activation function* after layer $^{(2)}$ and $^{(3)}$, where the CancelOut layer with the sigmoid activation function σ is utilized as input layer (superscript $^{(1)}$). Note that bias terms in FC layers are omitted for simplicity reasons. Then the output of the network is given by:

$$\hat{y} = \boldsymbol{X} \odot \sigma(W_{CO}) \cdot W^{(2)} w_1^{(3)} \tag{2}$$

where $\boldsymbol{X} = [x_1, x_2, x_3]$, $W_{CO} \in \mathbb{R}^3, W^{(2)} \in \mathbb{R}^3$, and $w_1^{(3)} \in \mathbb{R}$. Note, $\cdot$ is the dot product between two vectors.

The Eq.2 can be seen as a linear regression model:

$$\hat{y} = X\theta_1 + \theta_0 \tag{3}$$

where θ_1:

$$\theta_1 = \sigma(W_{CO}) \cdot W^{(2)} w_1^{(3)} \tag{4}$$

In case of multiple artificial neurons in the hidden layer$^{(2)}$(Fig. 2), the output is given by:

$$\hat{y} = \boldsymbol{X} \odot \underbrace{\sigma(W_{CO}) \cdot W_1^{(2)} w_1^{(3)}}_{\theta_1} + \boldsymbol{X} \odot \underbrace{\sigma(W_{CO}) \cdot W_2^{(2)} w_2^{(3)}}_{\theta_2} \tag{5}$$

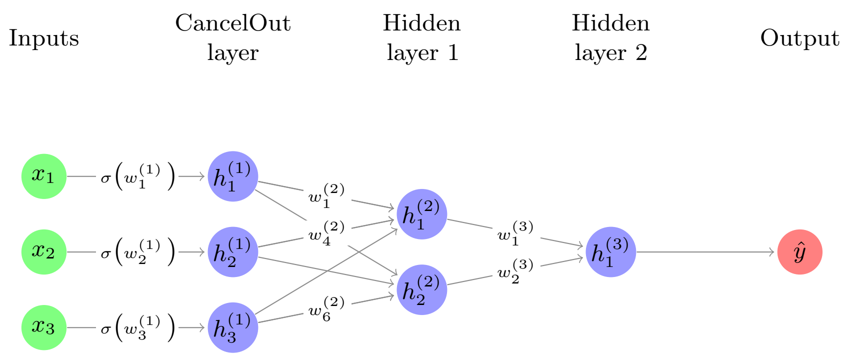


Fig. 2: A deep neural network with a CancelOut layer and two artificial neurons in the third layer.

From the Eq. 2 it can be seen that if the any value after the activation function $g(w_{CO}^{(1)}))$ in the CancelOut layer equals 0, than the correspoding input value x_n to $g(w_{CO}^{(1)}))$ is also 0. The following lemma summarizes this idea.

Lemma 1. *If a value after an activation function from the CancelOut layer is 0, a corresponding input variable does not affect the output of an ANN.*

As an illustration: let a value after the activation function from the CancelOut layer be 0.

$$\sigma(w_1^{(1)}) = 0 \tag{6}$$

Then $\hat{y}$ from Eq.5 from the ANN 5, can be seen as:

$$\begin{aligned}\hat{y} &= \boldsymbol{X} \odot \sigma(W_{CO}) \cdot W_1^{(2)} w_1^{(3)} + \boldsymbol{X} \odot \sigma(W_{CO}) \cdot W_2^{(2)} w_2^{(3)} = \\ &= \cancel{x_1\sigma(w_1^{(1)})w_1^{(2)}w_1^{(3)}} + x_2\sigma(w_2^{(1)})w_2^{(2)}w_1^{(3)} + x_3\sigma(w_3^{(1)})w_3^{(2)}w_1^{(3)} + \\ &\quad + \cancel{x_1\sigma(w_1^{(1)})w_4^{(2)}w_2^{(3)}} + x_2\sigma(w_2^{(1)})w_5^{(2)}w_2^{(3)} + x_3\sigma(w_3^{(1)})w_6^{(2)}w_2^{(3)}\end{aligned} \tag{7}$$

Remark 1. Clearly, if $w_1^{(2)} = w_4^{(2)} = 0$ in the Eq. 7 and the bias[4] term is also zero, then a CancelOut weight $w_1^{(1)}$ does not represent beneficial information. In order to avoid this outcome, we suggest to consider these recommendations:

– a proper choice of activation function in a layer after the CancelOut layer helps to bypass that issue;
– regularization terms can be used in a loss function;

[4] The bias term is omitted here, see Subsection 3.1.

– finally, it is advisable to have numerous artificial neurons in the layer after the CancelOut layer, since it diminishes the chance that all neurons weights in the layer after $W^{(2)}$ the CancelOut layer be zeros.

Moreover, if a weight w_i in the CancelOut layer is 0, then the gradient with respect to w_i is 0. The following lemma summarizes this idea.

Lemma 2. *If a value after an activation function from the CancelOut layer is approximately 0, then the gradient of the weight is also approximately 0.*

Combining Lemma 1.1 and Lemma 1.2, we get the following theorem:

Theorem 1. *Values after the activation function in the CancelOut layer indicate contributions to the output of a corresponding variable.*

Consequently, the CancelOut layer ranks features in a similar way linear models do, e.g. the large the absolute values in the CancelOut layer the more a corresponding input variable contributes to the output. Also, compared to linear models, the CancelOut method takes into account linear and non-linear combinations of input data.

Remark 2. A zero coefficient in CancelOut values $\sigma(W_{CO}^{(1)})$ leads to fewer optimization parameters, hence, a model learns faster. This also helps to reduce, it helps to reduce the number of features, therefore mitigating overfitting. Moreover, feature selection with the CancelOut layer can be adopted in two scenarios; a user can either specify the number of features or extract features using a chosen threshold.

Our FR approach is similar to [9], but in our work, the input scalar $\sigma(W_{CO}^{(1)})$ is bound in the chosen interval (e.g. for the sigmoid activation function is $\big(0,1\big)$). Therefore our approach is more stable, and it is simpler to rank features since a user selects only a threshold. Besides, the CancelOut FR method does not require a penalty coefficient in a loss function.

3.2 CancelOut layer weights initialization

A random weight initialization is not desired for the CancelOut layer, since it may give an advantage for one subset of features over another. Therefore, weights are initialized with uniformly distribution [5] with additional β coefficient:

$$W_{CO} \sim \mathcal{U}(-\frac{1}{\sqrt{n_{in}}} + \beta, \frac{1}{\sqrt{n_{in}}} + \beta) \tag{8}$$

where n_{in} is the size of the previous layer, and β is the coefficient which depends on the choice of an activation function.

We introduced β coefficient into Eq. 8 in order to control the initial output values after an activation function $g(W_{CO})$, it needs to be $g(W_{CO}) \neq 0$, because we assume that every feature is equally important e.g. for the logistic activation function $\beta \in [-3, \inf)$.

3.3 Loss function

In order to accelerate the feature ranking process in the CancelOut layer, we introduce two regularization terms in a loss function (Eq. 9):

$$\mathbb{L}(X, Y) = \mathcal{L}(X, Y) - \lambda_1\, var(\frac{W_{CO}}{N_v}) + \lambda_2 \left\| \frac{W_{CO}}{N_v} \right\|_1 \tag{9}$$

where $\mathcal{L}$ is a selected loss function, for the classification task it can be seen as:

$$\mathcal{L}_{CE}(X, Y) = -\frac{1}{n}\sum_{i=1}^{n}(y^{(i)} \ln \psi(x^{(i)}) + \left(1 - y^{(i)}\right) \ln \left(1 - \psi(x^{(i)})\right)) \\ - \lambda_1\, var(\frac{W_{CO}}{N_v}) + \lambda_2 \left\| \frac{W_{CO}}{N_v} \right\|_1 \tag{10}$$

where $X = \left\{x^{(1)}, \ldots, x^{(n)}\right\}$ is the set of input examples in the training dataset, and $Y = \left\{y^{(1)}, \ldots, y^{(n)}\right\}$ is the corresponding set of labels. The $\psi(x)$ represents the output of the neural network given input x, λ_1 and λ_2 are user-specified parameter coefficients $\lambda_1 \in [0, 1], \lambda_2 \in [0, 1]$, N_v is a number of variables in a dataset, and W_{CO} are CancelOut weights.

The mean square error (MSE) loss can be utilized for regression tasks:

$$\mathcal{L}_{MSE}(X, Y) = \frac{1}{2n}\sum_{i=1}^{n}(y^{(i)} - \psi(x^{(i)}))^2 - \lambda_1\, var(\frac{W_{CO}}{N_v}) + \lambda_2 \left\| \frac{W_{CO}}{N_v} \right\|_1 \tag{11}$$

The variance of the weights from the CancelOut layer $var(\frac{W_{CO}}{N_v})$ helps to stimulate diversity in the CancelOut layer, there $l1$ norm is used to introduce sparsity in W_{CO} weights and to constrain the variation to small weights. Also, $l1$ penalty restricts the model from selecting correlated features. Lastly, our feature selection approach supports all losses and does not require the realization terms.

Table 1: Datasets

Dataset	Samples	Features	Target
Statlog (Australian Credit Approval)	690	14 (continuous, nominal)	binary
Diabetes	442	10 (continuous, nominal)	regression
MNIST	70.000	784 (continuous)	multiclass

4 Experimental Results

In this section, we perform several experiments to evaluate different aspects of our CancelOut layer. In a first experiment, we examine the performance of our algorithm for classification and regression tasks, using the Statlog (Australian Credit Approval) and Diabetes dataset [3] (Sec. 4.1). We choose these datasets, because they contain different feature types such as continuous and nominal. We compare the proposed features from a CancelOut NN with a Random Forest and a Gradient Boosting algorithm using k-fold cross-validations (stratified for the classification experiment).

In a second experiment, we add a dummy variable $(Y + \epsilon_1) \in X$ with normally distributed noise ϵ to the Australien Credit Approval dataset, in order to see if the proposed method is able to detect a feature that is highly correlated with the target feature (Sec. 4.2). Additionally, we introduce a "noisy" variable $X_{random} \sim N(0,1) + \epsilon_2$ to assess, whether CancelOut discards irrelevant features. Note, $\epsilon_1 \neq \epsilon_2$.

Next, we compare feature importance characteristics from LASSO, SHAP [10], and CancelOut (Sec. 4.3). In a final experiment, we evaluate our model for the unsupervised scenario using a convolutional autoencoder (Sec. 4.4).

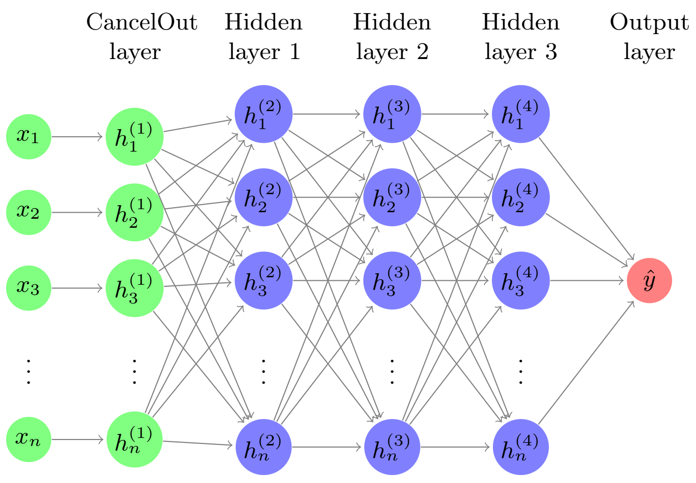
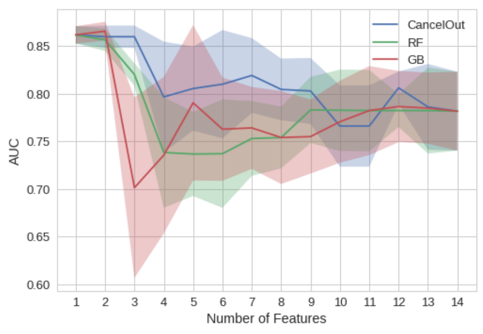


Fig. 3: A deep neural network architecture used for the experiments, where n is a number of variables.

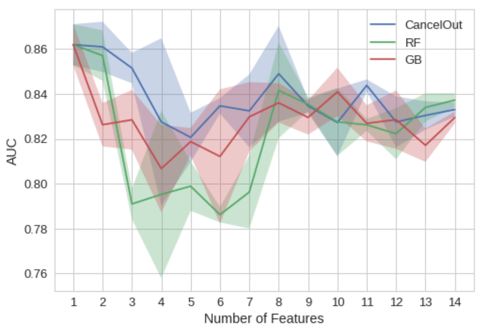
In all experiments, we use a five layers DL model (Fig. 3) where the input layer is the CancelOut layer with the sigmoid activation function after each FC layer the ReLU activation function [5] was applied. Further, we use the optimization algorithm Adam [7] with learning rate 0.003, β_1=0.9, β_2=0.999, and $\epsilon = 10^{-9}$. We utilize the early stopping technique to control overfitting of our model.

4.1 Feature Ranking

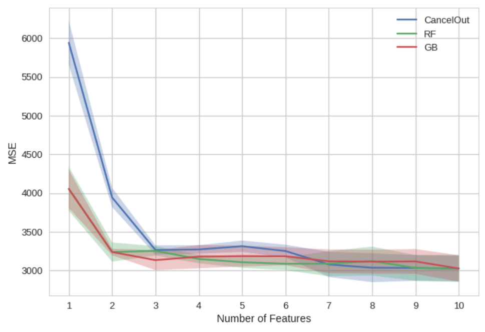
Classification example We illustrate the AUC scores for different sizes of feature subsets [3] in Fig. 4. The results are obtained by five-fold stratified cross-



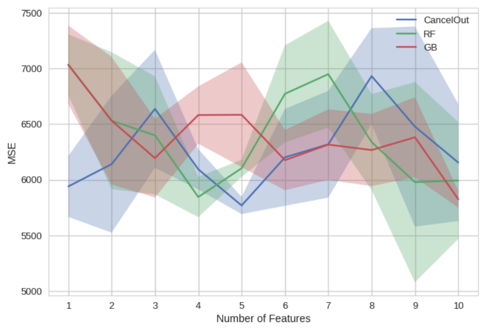
(a) Naive Bayes



(b) Decision Trees



(c) Linear Regression



(d) Regression Decision Trees

Fig. 4: A comparison of FS methods using Naive Bayes classifier (a) and decision trees (b) for the classification problem, and using linear regression (c) and decision trees regression (d) algorithms for the regression problem.

validation on the Australian Credit Approval dataset using Naive Bayes (Fig. 4a) and decision trees (Fig. 4b). Our algorithm achieves consistently good predictions for both classifiers and all feature set sizes. Moreover, CancelOut obtains superior predictions for small feature subsets. The variability in AUC is similar for all algorithms.

Regression example To evaluate CancelOut in context of a regression problem, we apply linear regression (Fig. 4c) and decision trees regression (Fig. 4d) on the diabetes dataset. We illustrate the MSE for different sizes of the reduced feature set in Fig. 4. We obtain the MSE scores again by five-fold cross-validation. CancelOut has disadvantages for linear regression if the number of features is smaller than three. However, our algorithm obtains competitive results for the mid- and end-range of the reduced feature set size. The error measures for regression tasks with decision trees highly fluctuate with the number of selected features. Yet, our algorithm obtains best results for a small reduced feature set.

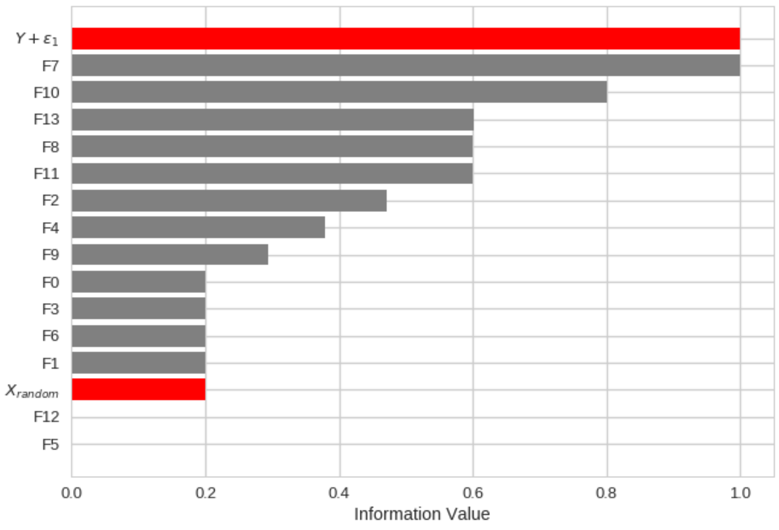


Fig. 5: A feature importance analysis for the Australian credit approval dataset [3] with two new features.

These observations suggest that CancelOut can generally obtain feature sets that perform well in regression tasks.

4.2 Identifying target and noisy features

In this experiment, we introduce two new features into the Australian Credit Approval dataset [3]. The first variable $Y + \epsilon_1$ is highly correlated to the target feature and the second is a random noise feature generated from the normal distribution $X_{random} \sim N(0, 1) + \epsilon_2$. The idea of the experiment is to show the ability of the proposed FR method to detect key and noisy features in the dataset.

In Fig. 5, we present a feature importance analysis for the augmented Australian Credit Approval dataset. The depicted values are the average of ten runs of an ANN obtained using the CancelOut layer. The analysis indicates that our method can successfully detect variables that are highly correlated to the target by evaluating them as the most important variable. Moreover, CancelOut mitigates the influence of noisy features by giving them low weights. This is shown exemplary by the low rank of X_{random}.

4.3 Evaluating individual feature importance

We investigated several feature analysis methods for the diabetes dataset and summarized it into Fig. 6. The purpose of this comparison is to show that CancelOut behaves comparable to other algorithms. Although there are differences in feature importance for the single features *age*, *sex*, *s2* and *s3*, the overall

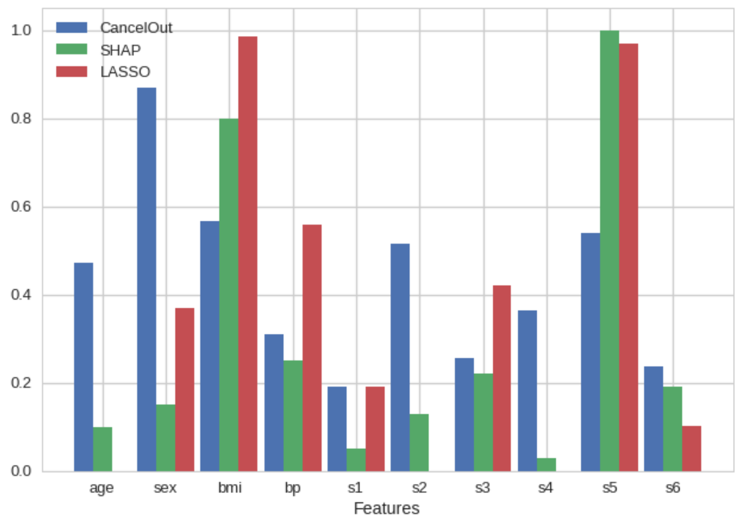


Fig. 6: A feature importance analysis for the diabetes dataset [3] using the proposed method (CancelOut), LASSO, and SHAP.

distribution of CancelOut weights is comparable to that of the SHAP and LASSO models.

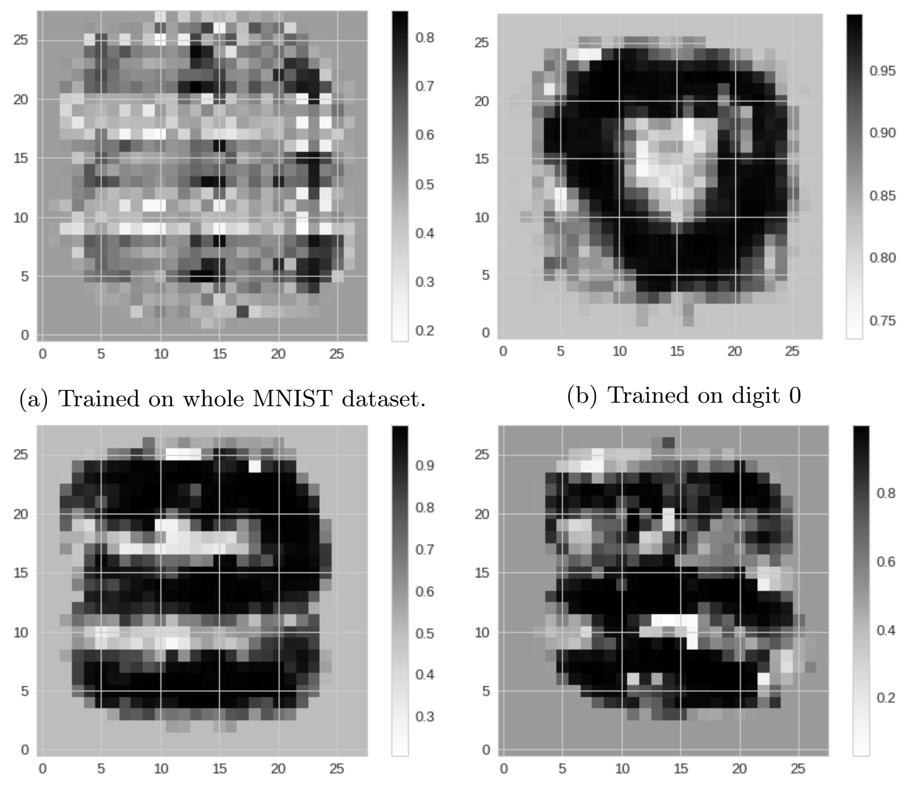
4.4 Unsupervised feature ranking using autoencoder

In this subsection, we demonstrate how the CancelOut layer can be utilized for unsupervised learning tasks with a convolutional autoencoder [11]. The architecture of the autoencoder consists of three convolutional neural network (CNN) layers in encoder and decoder parts, and the CancelOut as an input layer for the encoder. For this experiment, the MNIST dataset [8] is used.

Fig. 7 shows CancelOut variable weights after training the convolutional autoencoder on the whole dataset (a), only on digit 0 (b), only on digit 3 (c), and only on digit 8 (c). CancelOut captures the most relevant regions of the picture for all four training sets. The information provided by CancelOut layer weights can help in model understanding, debugging, and adjustment, e.g. by introducing a "focus" on relevant features if a model performs poorly.

5 Conclusion

In this paper, we introduced a novel feature ranking method using deep neural networks for various machine learning problems. The proposed method is extremely easy to implement, it can be done using all modern DL frameworks, and this method can be simply scaled. Due to the power of the neural networks, the presented approach learns linear and non-linear data dependencies. Also, the



(a) Trained on whole MNIST dataset.

(b) Trained on digit 0

(c) Trained on digit 3

(d) Trained on digit 8

Fig. 7: $\sigma(W_{CO})$ values from the CancelOut layer after training using MNIST dataset: (a) the whole dataset, (b) images from 0 class, (c) images from 3 class, (c) images from 8 class.

CancelOut layer can be applied for any data type and machine learning tasks, such as classification and regression problems or even for unsupervised problems as an input layer for an autoencoder. Finally, the proposed layer helps understand the data and its influence on the performance of DL models.

References

1. Chang, C.H., Rampasek, L., Goldenberg, A.: Dropout Feature Ranking for Deep Learning Models. arXiv e-prints (2017)
2. Cybenko, G.: Approximation by superpositions of a sigmoidal function. Mathematics of Control, Signals and Systems **2**(4), 303–314 (1989). https://doi.org/10.1007/BF02551274. URL `https://doi.org/10.1007/BF02551274`
3. Dua, D., Graff, C.: UCI machine learning repository (2017). URL `http://archive.ics.uci.edu/ml`

4. Gilpin, L.H., Bau, D., Yuan, B.Z., Bajwa, A., Specter, M., Kagal, L.: Explaining Explanations: An Overview of Interpretability of Machine Learning. arXiv e-prints (2018)
5. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016). `http://www.deeplearningbook.org`
6. Kasneci, G., Gottron, T.: Licon: A linear weighting scheme for the contribution ofinput variables in deep artificial neural networks. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, CIKM '16, pp. 45–54. ACM, New York, NY, USA (2016). https://doi.org/10.1145/2983323.2983746. URL `http://doi.acm.org/10.1145/2983323.2983746`
7. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. arXiv e-prints (2014)
8. LeCun, Y., Cortes, C.: MNIST handwritten digit database (2010). URL `http://yann.lecun.com/exdb/mnist/`
9. Li, Y., Chen, C.Y., Wasserman, W.W.: Deep feature selection: Theory and application to identify enhancers and promoters. Journal of Computational Biology **23**(5), 322–336 (2016). https://doi.org/10.1089/cmb.2015.0189. URL `https://doi.org/10.1089/cmb.2015.0189`. PMID: 26799292
10. Lundberg, S.M., Lee, S.I.: A Unified Approach to Interpreting Model Predictions. In: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (eds.) Advances in Neural Information Processing Systems 30, pp. 4765–4774. Curran Associates, Inc. (2017). URL `http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf`
11. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. chap. Learning Internal Representations by Error Propagation, pp. 318–362. MIT Press, Cambridge, MA, USA (1986). URL `http://dl.acm.org/citation.cfm?id=104279.104293`
12. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research **15**, 1929–1958 (2014). URL `http://jmlr.org/papers/v15/srivastava14a.html`
13. Zhang, Q., Nian Wu, Y., Zhu, S.C.: Interpretable Convolutional Neural Networks. arXiv e-prints (2017)

A.2.2 A Robust Unsupervised Ensemble of Feature-Based Explanations using Restricted Boltzmann Machine

Publication: Published in the 1st Workshop on eXplainable AI approaches for debugging and diagnosis at Conference on Neural Information Processing Systems (NeurIPS) 2021.

Contribution: I had an idea for a novel method of combining local explanations in order to achieve more robust and stable results. Along with Johannes Meier and Johan van den Heuvel, we implemented this framework using Python. Hamed Jalali aided in the theoretical formalization of our work, while Gjergji Kasneci provided valuable feedback on the structure of our paper. All of the co-authors, including myself, actively participated in the revision process to ensure the quality of the final manuscript.

A Robust Unsupervised Ensemble of Feature-Based Explanations using Restricted Boltzmann Machines

Vadim Borisov[*] University of Tübingen

Johannes Meier[†] University of Tübingen

Johan van den Heuvel[†] University of Tübingen

Hamed Jalali University of Tübingen

Gjergji Kasneci University of Tübingen

Abstract

Understanding the results of deep neural networks is an essential step towards wider acceptance of deep learning algorithms. Many approaches address the issue of interpreting artificial neural networks, but often provide divergent explanations. Moreover, different hyperparameters of an explanatory method can lead to conflicting interpretations. In this paper, we propose a technique for aggregating the feature attributions of different explanatory algorithms using Restricted Boltzmann Machines (RBMs) to achieve a more reliable and robust interpretation of deep neural networks. Several challenging experiments on real-world datasets show that the proposed RBM method outperforms popular feature attribution methods and basic ensemble techniques.

1 Introduction

As the applications of deep neural networks (DNNs) continue to grow, the black-box nature of DNNs creates potential trust issues [1]. Moreover, numerous life-critical (such as medical, automotive, or financial) applications utilize DNNs for various estimation tasks. In such applications, and especially for the long-term acceptance of artificial intelligence (AI) solutions, a deeper understanding and trust in the produced results is crucial. Furthermore, feature attribution methods are important tools for deep model debugging and diagnosis [2].

Explaining how the input influences the output for a given DNN is one form to interpret the black-box nature of the DNN and bring trust to a system. These so-called feature-based explanation methods received a lot of attention in recent years [1, 3–5]. They can be grouped into three broad categories, (1) approaches based on gradient information [6, 7], (2) perturbation-based approaches [8, 9], and (3) attribution-based approaches [3, 10]. Interestingly, different feature-based explanation approaches regularly produce mixed views on the main attributes (areas of an image or variables), and in the absence of the ground truth, it is still a challenge to verify which explanation method is the most trustworthy. Moreover, in the AI community, there are no yet accepted quality measures for feature-based explanations. *All these difficulties resulted in a large number of different explanation methods and in a lack of consensus on which techniques are most reliable.*

Within the machine learning (ML) community, there is much work on the combination of methods that do not always agree with each other, i.e. *ensemble learning* [11, 12]. Normally ensemble models outperform the non-ensemble models and turn out to be more robust to outliers. The main idea is that if multiple methods make mistakes in different areas, combining them in an intelligent way improves

[*]Corresponding author: `vadim.borisov@uni-tuebingen.de`

[†]Equal contribution

performance and reduces the effect of outliers as compared to the single method. Moreover, from statistical learning theory and practical applications, it is well understood that ensemble learning is the path of choice towards a more robust machine learning system [13], even in unsupervised learning scenarios where the target is not available [14, 15].

In this work, utilizing ideas from [15, 11] and [16, 17], we introduce a novel approach for the unsupervised ensemble learning of reliable and robust feature-based explanations for deep neural networks. To this end, we propose using a model based on Restricted Boltzmann Machines (RBMs), which achieves this goal by aggregating the results (saliency maps) of different feature-based explanation methods in a principled probabilistic fashion. Also, it has been shown that an RBM can be used in the truth discovery setting [18, 19], which is analogous to our task of finding a reliable feature importance map from different importance maps.

The main contributions of this work are:

- We introduce a novel method for a robust and reliable feature-based explanation using ensemble learning.
- We empirically and visually show the superior performance of the proposed method in comparison to state-of-the-art feature attribution baselines.
- We open-source our code and make it publicly available, as an RBM ensemble framework: (https://github.com/JohanvandenHeuvel/AggregationOfLocalExplanations), Besides, we also developed a single Python package with various evaluation metrics for feature attribution methods metrics: https://github.com/meier-johannes94/ExplainableAIImageMeasures

The paper is organized as follows: In Section 2 we discuss the related work and provide essential background information. Section 3 presents the proposed ensemble method for local feature-based explanations using an RBM. In Section 4, we present the results of various experiments. Section 5 discusses limitations of our work and ways to address them in the future. Section 6 concludes our work with a short summary.

2 Background and Related Work

This part of the manuscript provides the needed background and discusses related approaches. First, we present the basic notation used in this work and proceed by presenting two ensemble techniques for aggregating feature importance maps.

2.1 Feature Attribution Function

Formally, a feature attribution function can be seen as $\phi(f, \mathbf{x}, c_x)$, where f is a black box model and $\mathbf{x}$ is an input data point from a corresponding class c_x. The output of ϕ is an explanation vector or matrix $\mathbf{e}_{f(\mathbf{x})}$, where each element of $\mathbf{e}_{f(\mathbf{x})}$ is an importance score for the corresponding feature value in $\mathbf{x}$. A large positive or negative value in $\mathbf{e}_{f(\mathbf{x})}$ indicates that the corresponding feature (pixel) has a large influence on the outcome of the black-box model f.

Assumption 2.1 In the following, we assume that a <u>true</u> feature attribution $\bar{\mathbf{e}}_f(\mathbf{x})$ for a given model f and input $\mathbf{x}$ exists and can be constructed by adequately aggregating available attributions $\mathbf{e}_{f(\mathbf{x}),i}, i \in \{1, ..., N\}$, where N is the number of baseline explanations (from N baseline methods).

For better readability and simplicity, from here we omit the index $f(\mathbf{x})$.

The goal of any explanation method ϕ is to obtain an attribution $\mathbf{e}$ that is as close as possible to $\bar{\mathbf{e}}$. Note that our method naturally generalizes to probabilistic local explanation methods [20]. Given the before-mentioned assumption, we can say that there is a joint probability distribution of the pair $(\mathbf{e}, \bar{\mathbf{e}})$ parametrized by θ.

$$p_\theta(\mathbf{e}, \bar{\mathbf{e}}) = p_\theta(\bar{\mathbf{e}})p_\theta(\mathbf{e}|\bar{\mathbf{e}}).$$

The joint distribution $p_\theta(\mathbf{e}, \bar{\mathbf{e}})$ is not known, and neither are the marginals $p_\theta(\mathbf{e})$, $p_\theta(\bar{\mathbf{e}})$.

For the following theoretical results we require that the explanation methods give independent explanations when conditioned on the true explanation. However, as with Naive Bayes methods, for practical purposes, this assumption can be violated without negatively impacting the aggregation quality [15]. Also note that we do assume some consistency between the explanations, following the assumption that feature attributions reflect the underlying (but unknown) importance distributions of the feature values [21].

Assuming conditional independence between the provided baseline explanations given the (unknown) true explanation, we have

$$p_\theta(\mathbf{e}|\bar{\mathbf{e}}) = \prod_{n=1}^{N} p_\theta(\mathbf{e}_n|\bar{\mathbf{e}}),$$

where $\mathbf{e}_n$ is a baseline explanation in the ensemble involving N different baseline explanations.

2.2 Ensemble Learning

As we state in the introduction, ensemble learning is a well-studied approach for improving the performance of an ML system. One of the most basic ensemble methods employs the mean of results of base learners [13], where a *base learner* is a single algorithm from the ensemble.

$$\mathbf{e}_{mean} = \frac{1}{N}\sum_{n=1}^{N} \mathbf{e}_n. \tag{1}$$

A significant drawback of the *mean ensemble approach* is that it still is sensitive to outliers or noisy estimations. Furthermore, data scaling may strongly influence the aggregation. In Section 4, these weaknesses of the *mean ensemble approach* are also seen in the experimental evaluation.

To mitigate these weaknesses, the authors of [22] propose to take the local uncertainty into account. To this end, they divide the mean by the local variance plus a constant ϵ for stability reasons, which results in the *variance ensemble approach*:

$$\mathbf{e}_{var} = \frac{1}{N}\sum_{n=1}^{N} \frac{\mathbf{e}_n}{\sigma_*(\mathbf{e}_{i\in\{1,\ldots,N\}}) + \epsilon},$$

where $\sigma_*(\mathbf{e}_{i\in\{1,\ldots,N\}})$ is the point-wise standard deviation over all the available explanations $\mathbf{e}_i, i \in \{1, ..., N\}$. This method assigns less relevance to explanations that have high disagreement with the remaining explanations.

Also, the authors of [23] proposed a novel method to aggregate Shapley values through an explanation function that minimizes sensitivity.

3 Ensemble Learning using Restricted Boltzmann Machines

In this section, we present an unsupervised aggregation of feature attribution maps using a Restricted Boltzmann Machine (RBM). Similar aggregation techniques have been proposed in other contexts, e.g., in [15, 18].

3.1 The Restricted Boltzmann Machine

An RBM is an undirected bipartite graph that can be parametrized by a neural network. It is a variant of the Boltzmann Machine, with the additional property that there are no connections within both the group of visible nodes or the group of hidden nodes. The advantage of this property is that nodes in one group are conditionally independent of each other, given that we know the state of the nodes in the other group. One of the main characteristics of an RBM is that it can learn a probability distribution over its set of inputs. A graphical representation of an example RBM is shown in Figure 1.

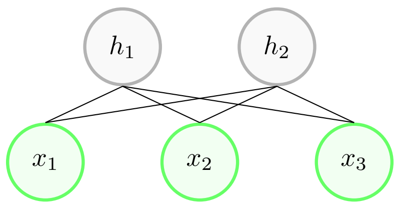


Figure 1: An RBM with three visible and two hidden units. In our work, we use an RBM with a single hidden node.

The formal definition of an RBM is as follows. There is a set X of n visible binary random variables and a set H of m hidden binary random variables. The RBM has parameters $\lambda = (\mathbf{W}, \mathbf{a}, \mathbf{b})$. $\mathbf{W}$ is the weight matrix of the connections between the nodes, $\mathbf{a}$ is the bias of the visible layer and $\mathbf{b}$ is the bias of the hidden layer. Each possible state of the RBM, i.e. the particular values of (X, H), is associated with the following energy function (in matrix notation):

$$E_\lambda(\mathbf{x}, \mathbf{h}) = -(\mathbf{a}^T\mathbf{x} + \mathbf{b}^T\mathbf{h} + \mathbf{x}^T\mathbf{W}\mathbf{h}),$$

which then can also be used to define the joint probability distribution for the visible and hidden vectors is defined in terms of the energy function:

$$P_\lambda(\mathbf{x}, \mathbf{h}) = \frac{1}{Z}e^{-E(\mathbf{x},\mathbf{h})},$$

where Z is the sum over $e^{-E(\mathbf{x},\mathbf{h})}$ for all possible configurations $\mathbf{x}, \mathbf{h}$, which can be seen as a normalization constant to ensure that all probabilities sum to 1, also known as the partition function.

The optimization objective of the RBM is to maximize the expected log probability of a training sample $\mathbf{x}$:

$$\begin{aligned} &\underset{\lambda}{\operatorname{argmax}}\, \mathbb{E}[\log P_\lambda(X = \mathbf{x})] = \\ &\underset{\lambda}{\operatorname{argmax}}\, \mathbb{E}[\log \sum_{\mathbf{h}} P_\lambda(X = \mathbf{x}, H = \mathbf{h})]. \end{aligned} \tag{2}$$

To train an RBM, a gradient-based optimization can be applied using the contrastive divergence algorithm [24, 25].

3.2 Aggregation of Local Explanations using an RBM

Given an RBM with N visible nodes and one hidden node, with input $\mathbf{x}$, where N is the number of baseline explanations in our ensemble, it can be shown that the true posterior probability of y can be efficiently estimated (Lemma 4.1, Lemma 4.2 from [15]). Furthermore, given the previously discussed mild assumptions on the input data (which are in line with those in [15]), the maximum likelihood estimate $\bar{\lambda}_{MLE}$ for the parameters of the RBM, the RBM posterior probability $P_{\bar{\lambda}_{MLE}}(H = 1|X = \mathbf{x})$ converges to true posterior $P_\theta(Y = 1|X = \mathbf{x})$.

Hence, we are able to apply the RBM to the *unsupervised aggregation* of N available feature-based explanations. We assume a joint distribution $p_\theta(\mathbf{e}, \bar{\mathbf{e}})$, and that the $\mathbf{e}_i$'s are conditionally independent from each other given $\bar{\mathbf{e}}$. By fitting the RBM we learn the parameters θ and thus obtain the relationship between our known explanations $\mathbf{e}_i$ and the true explanation $\bar{\mathbf{e}}$. The ensemble pipeline of the proposed method is depicted in Fig. 2.

In order to preserve the spatial information for visual data using the RBM-based ensemble, we do a pixel-wise aggregation. Therefore, for each pixel we train a Bernoulli RBM with a single hidden unit.

A known limitation of an RBM is the so-called *flipping issue* [15, 18, 19], which arises from the RBM parametrization symmetry. That is, the weights of the RBM can be flipped symmetrically without changing the behavior of the RBM. In order to avoid this unwanted effect, we propose two approaches: flip detection and metric optimization. The *flip detection* algorithm extends the idea from Remark 4.3 in [15], by comparing the top 5 % of most important and 5 % of less important pixels to the mean baseline. The algorithm inverts the current important scores if there is a strong disagreement between the proposed approach and the mean baseline. The *metric optimization* method utilizes the chosen metric to overcome the flipping issue. It compares two versions of the RBM ensemble results and selects the one with a better performance according to the selected metric.

4 Experiments

To demonstrate the effectiveness of the proposed ensemble algorithm we conduct various visual and quantitative experiments. First, we present the visual inspection results on the MNIST [27] and ImageNet [26] datasets in two settings, with and without noisy explanation maps in our ensemble.

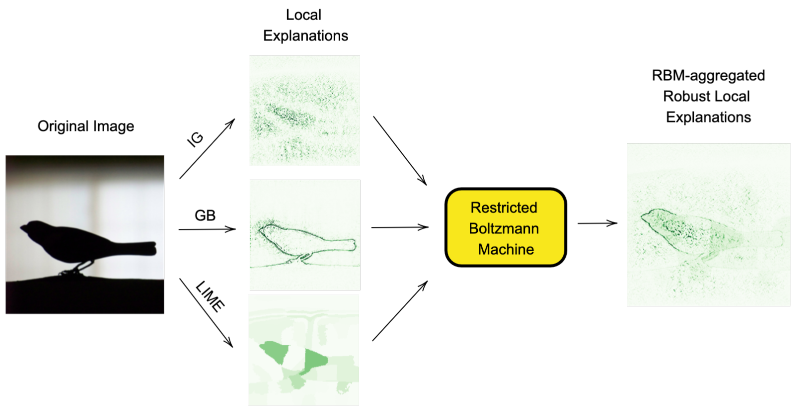

Original Image
Local Explanations
IG
GB
LIME
Restricted Boltzmann Machine
RBM-aggregated Robust Local Explanations

Figure 2: An overview on the ensemble of feature attribution maps from three different local explanation algorithms using an RBM for an image from the ImageNet dataset [26].

Despite a growing body of research focusing on explainable ML, the fair quantitative comparison of local explanation (or saliency-based) algorithms is still an open question, since the existing methods mostly utilize the pixel perturbation strategy (e.g., removing the most or least important pixels and reporting the change in recognition quality) [28, 29]. Also, such evaluations have a significant drawback, replacing image pixels with black or "mean" or any other pixel values may lead to artifacts affecting the data distribution [29, 30]. Nevertheless, since pixel perturbation analyses are employed in many related works, for our quantitative analysis we select the following approaches: the pixel perturbation for insertion (IAUC) and deletion (DAUC). Furthermore, we utilize the iterative removal of features (IROF) analysis [31]. We explain each evaluation method in detail in the corresponding subsections.

In our last experiment, we demonstrate that our ensemble approach can be also used within a singe feature attribution framework to achieve more robust and stable explanations. Since, it has been shown that hyperparameters choice can significantly affect the saliency maps [32].

4.1 Visual Inspection for Image Data

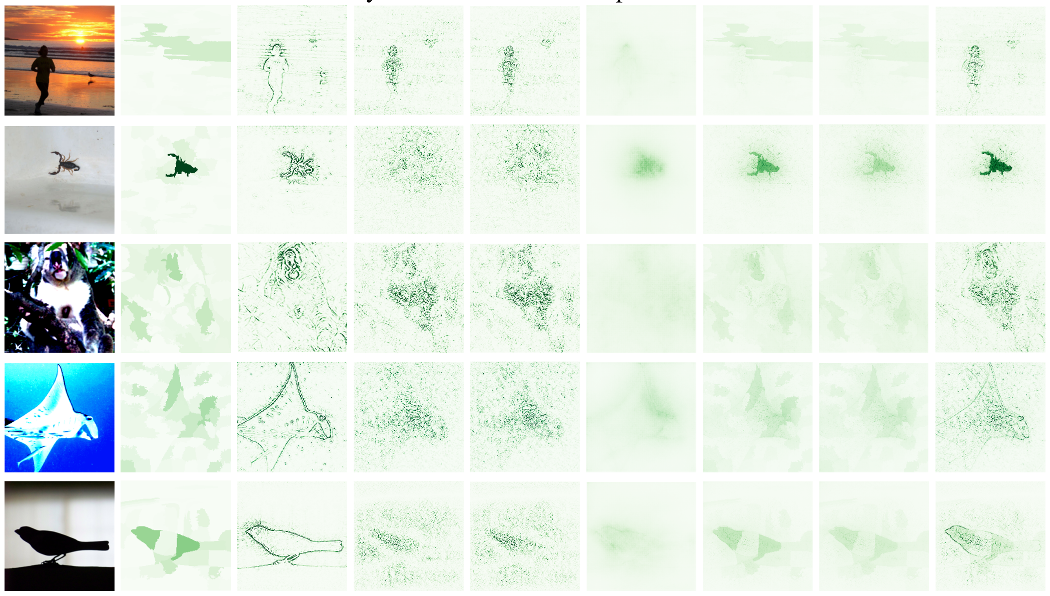
In our first experiment, a visual evaluation on images from ImageNet [26] and MNIST [27] is performed for several baseline and ensemble methods. We provide benchmark outcomes for two settings, with and without fifteen noisy baseline explanations in an ensemble. The results are depicted in Table 1.

Without artificial noise in the ensemble. We select four samples from ImageNet dataset [26] and five baseline explanation methods for the ensemble models: LIME [8], Guided Backpropagation (GB) [33], Integrated Gradients (IG) [34], Gradient SHAP (GS) [9], and SmoothGrad (SG) [35]. We compare the proposed RBM ensemble strategy to simple mean and variance ensembles [22]. The results in Table 1 show that our approach produces sharp and visually appealing saliency maps in comparison to other ensemble baselines. In comparison to the baseline explanation methods, the proposed ensemble technique seems to produce more reliable and robust results by highlighting commonalities among the baseline methods and by mitigating the noise coming from the single baseline methods.

With artificial noise in the ensemble. We challenge the discussed approaches by adding fifteen baselines with random noise sampled from the standard normal distribution $\mathbf{e}_{rand} \sim \mathcal{N}(0,\, 1)$ to the ensemble. The results in Table 1 reveal that the proposed RBM-based aggregation method mitigates noise and hence results in more robust saliency maps in comparison to the other ensemble baselines.

Original	LIME [8]	GB [33]	IG [34]	GS [9]	SG [35]	Mean ensemble	Variance ensemble	**RBM ensemble**

Without noisy feature attribution maps in the ensemble



With noisy feature attribution maps in the ensemble

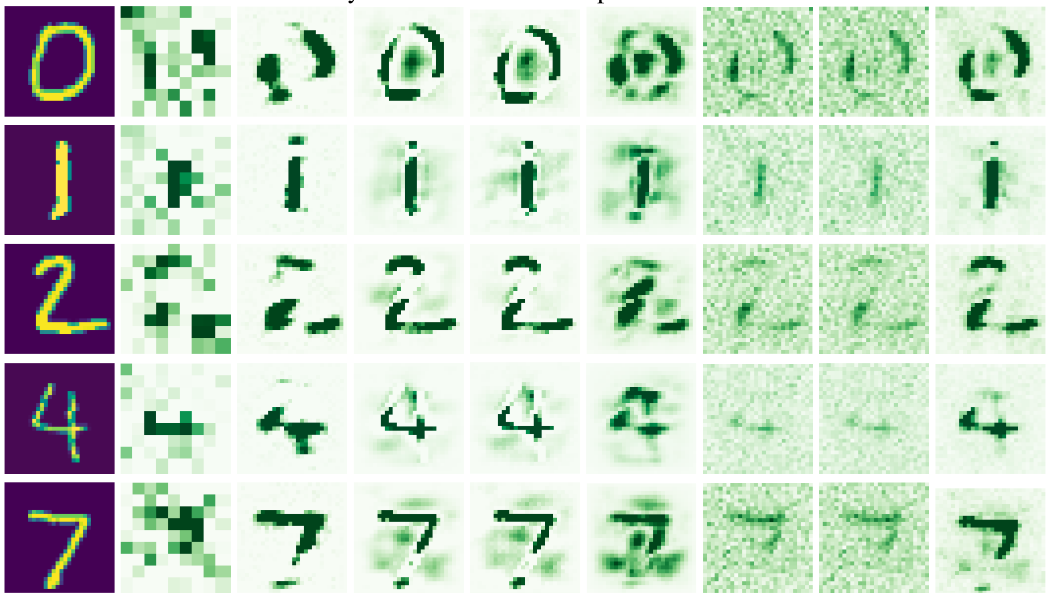


Table 1: A visual comparison between baseline methods and ensemble methods on ImageNet [26] and MNIST [27] datasets.

Method	Insertion (IAUC)	Deletion (DAUC)	IROF [31]
Gradient SHAP [9]	0.61 ± 0.42	0.22 ± 0.29	0.73 ± 0.24
DeepLIFT [3]	0.62 ± 0.42	0.23 ± 0.30	0.73 ± 0.23
LIME [8]	$\mathbf{0.80} \pm \mathbf{0.31}$	0.23 ± 0.23	$\mathbf{0.76} \pm \mathbf{0.22}$
Saliency map [38]	0.50 ± 0.35	0.37 ± 0.32	0.65 ± 0.25
SmoothGrad [35]	0.60 ± 0.26	0.38 ± 0.29	0.63 ± 0.26
Integrated Gradients [34]	0.66 ± 0.42	$\mathbf{0.19} \pm \mathbf{0.27}$	0.75 ± 0.23
Guided Backpropagation [33]	0.54 ± 0.38	0.49 ± 0.36	0.65 ± 0.25
Original Image	0.52 ± 0.32	0.53 ± 0.34	0.47 ± 0.30
Mean Ensemble	$\mathbf{0.79} \pm \mathbf{0.33}$	0.25 ± 0.28	0.70 ± 0.26
Variance Ensemble [22]	0.62 ± 0.36	0.39 ± 0.31	0.71 ± 0.26
RBM ensemble with the flip detection	0.76 ± 0.38	0.19 ± 0.26	0.76 ± 0.22
RBM ensemble with the metric optimization	0.77 ± 0.37	$\mathbf{0.18} \pm \mathbf{0.24}$	$\mathbf{0.76} \pm \mathbf{0.22}$

Table 2: A quantitative comparison between single and ensemble methods for the pixel perturbation: IAUC (higher is better), DAUC (lower is better), and IROF (higher is better) experiments on 10,000 samples from the CIFAR10 validation dataset [36].

4.2 Pixel Perturbation Experiment

In the first quantitative experiments, we compare multiple baseline models and ensemble methods on the CIFAR10 dataset [36] by removing the most important pixels (according to a scoring function) and reporting the area under a curve score (DAUC). In addition, we also follow the approach of inserting the most important pixels into an empty image and again report the area under a curve (IAUC). Thus, an ideal feature scoring function has a large IAUC and low DAUC. These benchmark methods well accepted by the research community [37]. For this experiment, we select the following algorithms as baseline explanation methods: Gradient SHAP [9], DeepLIFT [3], LIME [8], Saliency maps [38], SmoothGrad [35], Integrated Gradients [34], Guided Backpropagation [33]. As suggested in [11], we add the original image as a baseline to the ensemble. However, according to our experiments, adding th original image to the ensemble does improve the overall ensemble performance. We report all scores for the baseline approaches and the ensemble methods in Table 2.

4.3 IROF Experiment

In [22] the authors propose the IROF measure as an extension to the work [39]. The main idea of the IROF benchmark is as follows: the image is divided into superpixels using the SLIC algorithm [40]. Superpixels are regional blocks of pixels within an image where the contained pixels share a high similarity measure among each other. The relevancy for each superpixel is calculated by averaging over the attribution scores over all contained pixels (inside the superpixel). After, the superpixels are sorted descending by their relevancy. The entire superpixels are gradually replaced by a baseline and sent through the network again to measure the new recognition quality for the modified image wrt. to the target label. For more accurate attribution methods, the recognition quality decreases faster, and thus the area under the curve is lower. The IROF score is defined as the area over the curve (AOC): $AOC = 1 - AUC$. Higher values, therefore, indicate a better attribution. We use the same baseline methods as in the pixel perturbation experiment (Sec. 4.2). The results are listed in Table 2.

4.4 An RBM Ensemble Within a Singe Explanation Framework

In this experiment, we demonstrate that even for multiple baseline explanations of the same explanation method, the unsupervised ensemble with an RBM can lead to an improvement. To this end, we select the LIME [8] method with a different hyperparameter - the number of superpixels in the image. For the baselines (LIME-0, LIME-1, and LIME-2), we used 10, 100, and 1000 superpixels per image, respectively. The results can be seen in Fig. 4. The main idea is that each lime method has a different granularity level, thus highlighting distinct detail levels, and the proposed method's aggregation may help improve the reliability and robustness of feature attributions.

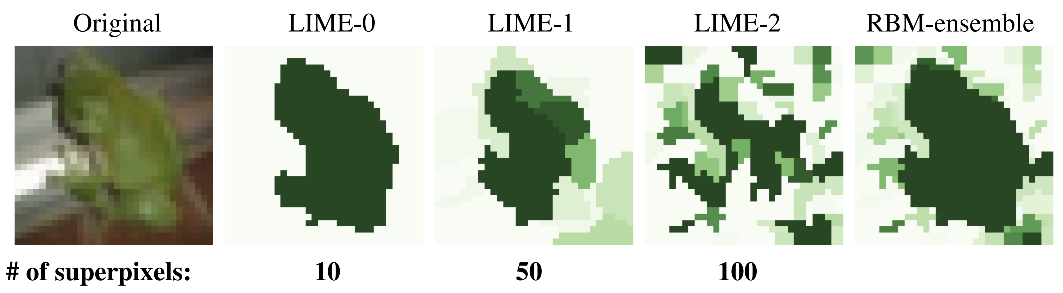


Figure 3: Three local feature attribution maps for a single data sample from CIFAR10 dataset [36] using the LIME algorithm [8] with different number of superpixels and the proposed RBM ensemble of the selected feature attribution maps.

4.5 Reproducibility

For reproducibility reasons, we describe the data preprocessing step used for all the experiments and provide information about packages used in this work. Also, the code for every experiment is publicly available online (see the links provided in Section 1).

To achieve a fair comparison, the image data from all datasets was preprocessed in the same way for each baseline. We performed a per saliency map normalization before the aggregation. In every experiment, we used the ResNet18 neural network architecture [41], except for the experiment on the MNIST dataset where we utilized a simple five layers convolutional neural network. We use a pre-trained model for ImageNet dataset [26] from torchvision library [42].

For the experiments we used the Bernoulli RBM implementation from the Scikit-Learn library [43] with following hyperparameters for each experiment: for the MNIST dataset we set the batch size to 5, the learning rate to 0.01, and the number of iterations to 100. For CIFAR10 and ImageNet datasets we use the following hyperparameters: a batch size of 35, a learning rate of 0.001, and a number of iterations is 250. The rest of hyperparameters are default to the scikit-learn package. For all baseline explanation techniques we use the publicly available open-source implementations from the captum library [44] with their default hyperparameters.

5 Discussion and Future Work

The results of multiple experiments with the proposed RBM ensemble show its competitive performance compared to base explanation techniques and other ensemble approaches. We hypothesize moderate performance of the RBM ensemble on the insertion (IAUC) benchmark is connected to our data preparation step since we filter the negative values for every saliency map in the ensemble.

The computational complexity of an ensemble method primarily depends on the base learner. In our case, the base explanation techniques are relatively fast, especially on specialized hardware (GPU or TPU), where an RBM has low computational complexity.

The gradient-based methods frequently produce noisy explanations. We empirically demonstrated that our approach reduces the noise in the final ensemble (Tab. 1). Therefore, we believe that the RBM aggregation of multiple saliency maps from gradient-based feature attributions is a powerful tool for improving the overall reliability of local explanations.

As part of our future work, we aim to evaluate our aggregation approach on larger datasets. Furthermore, methods for selecting a few quite reliable base explanations for aggregation might lead to efficient explanations ensembles for larger datasets.

Finally we expect that the proposed approach can be easily adapted to handle local explanations over structured tabular data, where the explanation of deep neural networks is an essential task for many crucial applications such as healthcare and finance [45].

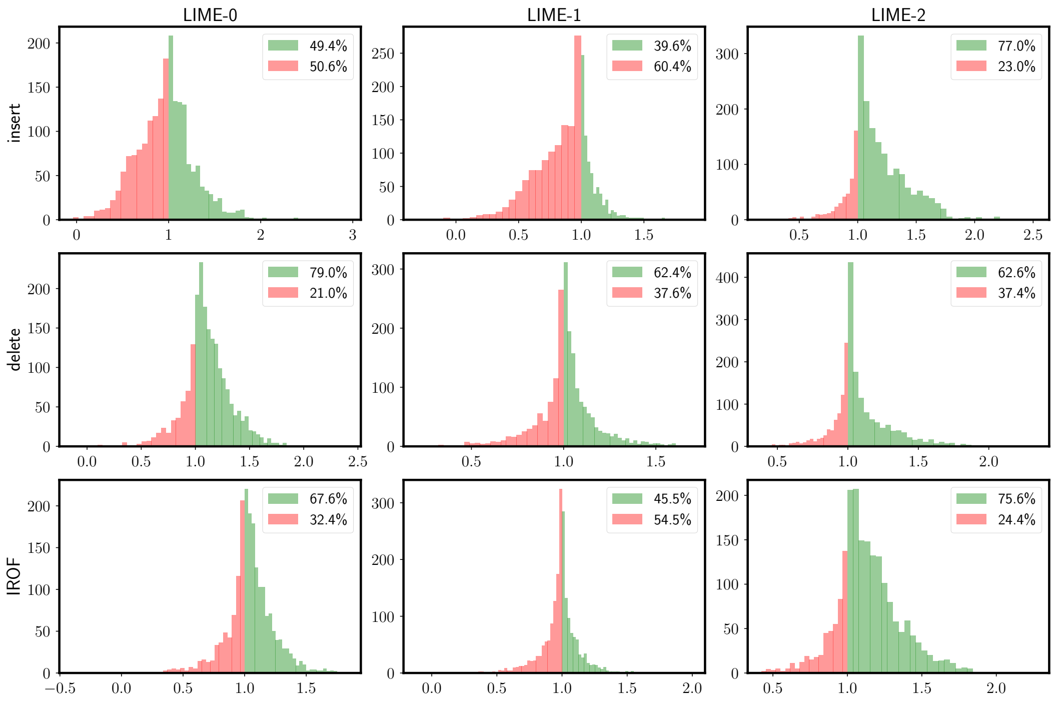


Figure 4: Distributions of differences where the proposed RBM ensemble shows better (green) and inferior (red) results in comparison to a baseline explanation technique according to insertion, deletion, and IROF metrics (score 1 means them being equal). The ensemble consists of the feature attributions from the same algorithm - LIME, but different hyperparameters. We randomly sampled 2000 images from CIFAR10 [36] for this experiment.

6 Conclusion

In this work, we presented a novel approach to unsupervised aggregation of feature-based explanations using Restricted Boltzmann Machines with the aim of reliably interpreting the influence of inputs on the output of deep neural networks. In addition to explanatory reasons, the latter is also essential for debugging and diagnostic purposes and serves the long-term acceptance of deep learning in real-world applications.

Using the proposed approach, we demonstrated through visual and quantitative experiments its ability to obtain more robust and reliable explanations than other existing ensemble methods. In a setting with noisy attribution maps in an ensemble, the proposed approach successfully selects only the valuable information, mitigating noise. Moreover, our work illuminates and mitigates the problem of possible contradictory results that may be obtained by different explanation and evaluation methods. Finally, we note that our approach can also be used within a single interpretability framework to reduce the sensitivity of a feature-based explanatory approach to its hyperparameters.

References

[1] Yu Zhang, Peter Tiňo, Aleš Leonardis, and Ke Tang. A survey on neural network interpretability. *arXiv preprint arXiv:2012.14261*, 2020.

[2] Mattia Setzu, Riccardo Guidotti, Anna Monreale, Franco Turini, Dino Pedreschi, and Fosca Giannotti. Glocalx-from local to global explanations of black box ai models. *Artificial Intelligence*, 294:103457, 2021.

[3] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International Conference on Machine Learning*, pages 3145–3153. PMLR, 2017.

[4] Leilani H Gilpin, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pages 80–89. IEEE, 2018.

[5] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. *arXiv preprint arXiv:1810.03292*, 2018.

[6] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pages 3319–3328. PMLR, 2017.

[7] Gjergji Kasneci and Thomas Gottron. Licon: A linear weighting scheme for the contribution ofinput variables in deep artificial neural networks. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 45–54, 2016.

[8] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[9] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Proceedings of the 31st international conference on neural information processing systems*, pages 4768–4777, 2017.

[10] Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. Layer-wise relevance propagation: an overview. *Explainable AI: interpreting, explaining and visualizing deep learning*, pages 193–209, 2019.

[11] Chun-Xia Zhang, Jiang-She Zhang, Nan-Nan Ji, and Gao Guo. Learning ensemble classifiers via restricted boltzmann machines. *Pattern Recognition Letters*, 36:161–170, 2014.

[12] Zhi-Hua Zhou. *Ensemble methods: foundations and algorithms*. Chapman and Hall/CRC, 2019.

[13] Ludmila I Kuncheva and Christopher J Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, 51(2):181–207, 2003.

[14] Ariel Jaffe, Ethan Fetaya, Boaz Nadler, Tingting Jiang, and Yuval Kluger. Unsupervised ensemble learning with dependent classifiers. In *Artificial Intelligence and Statistics*, pages 351–360. PMLR, 2016.

[15] Uri Shaham, Xiuyuan Cheng, Omer Dror, Ariel Jaffe, Boaz Nadler, Joseph Chang, and Yuval Kluger. A deep learning approach to unsupervised ensemble learning. In *International conference on machine learning*, pages 30–39. PMLR, 2016.

[16] Gjergji Kasneci, Jurgen Van Gael, David Stern, and Thore Graepel. Cobayes: bayesian knowledge corroboration with assessors of unknown areas of expertise. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 465–474, 2011.

[17] Gjergji Kasneci, Jurgen Van Gael, Ralf Herbrich, and Thore Graepel. Bayesian knowledge corroboration with logical rules and user feedback. In *Joint european conference on machine learning and knowledge discovery in databases*, pages 1–18. Springer, 2010.

[18] Klaus Broelemann, Thomas Gottron, and Gjergji Kasneci. Restricted boltzmann machines for robust and fast latent truth discovery. *arXiv preprint arXiv:1801.00283*, 2017.

[19] Klaus Broelemann and Gjergji Kasneci. A gradient-based split criterion for highly accurate and transparent model trees. *CoRR*, abs/1809.09703, 2018.

[20] Xingyu Zhao, Wei Huang, Xiaowei Huang, Valentin Robu, and David Flynn. Baylime: Bayesian local interpretable model-agnostic explanations, 2021.

[21] Pascal Sturmfels, Scott Lundberg, and Su-In Lee. Visualizing the impact of feature attribution baselines. *Distill*, 5(1):e22, 2020.

[22] Laura Rieger and Lars Kai Hansen. Aggregating explanation methods for stable and robust explainability, 2020.

[23] Umang Bhatt, Adrian Weller, and José MF Moura. Evaluating and aggregating feature-based model explanations. *arXiv preprint arXiv:2005.00631*, 2020.

[24] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[25] Yoshua Bengio. *Learning deep architectures for AI*. Now Publishers Inc, 2009.

[26] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[27] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[28] Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks, 2018.

[29] Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. Evaluating feature importance estimates. *CoRR*, abs/1806.10758, 2018.

[30] Johannes Haug, Stefan Zürn, Peter El-Jiz, and Gjergji Kasneci. On baselines for local feature attributions. *CoRR*, abs/2101.00905, 2021.

[31] Laura Rieger and Lars Kai Hansen. Irof: a low resource evaluation metric for explanation methods. *arXiv preprint arXiv:2003.08747*, 2020.

[32] Naman Bansal, Chirag Agarwal, and Anh Nguyen. Sam: The sensitivity of attribution methods to hyperparameters. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pages 8673–8683, 2020.

[33] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net, 2015.

[34] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. *CoRR*, abs/1703.01365, 2017.

[35] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

[36] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *citeseerx*, 2009.

[37] Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. *arXiv preprint arXiv:1806.07421*, 2018.

[38] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps, 2014.

[39] Wojciech Samek, Alexander Binder, Grégoire Montavon, Sebastian Bach, and Klaus-Robert Müller. Evaluating the visualization of what a deep neural network has learned. *CoRR*, abs/1509.06321, 2015.

[40] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. Slic superpixels. Technical Report 149300, EPFL, June 2010.

[41] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[42] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[43] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[44] Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, and Orion Reblitz-Richardson. Captum: A unified and generic model interpretability library for pytorch, 2020.

[45] Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. Deep neural networks and tabular data: A survey. *arXiv preprint arXiv:2110.01889*, 2021.

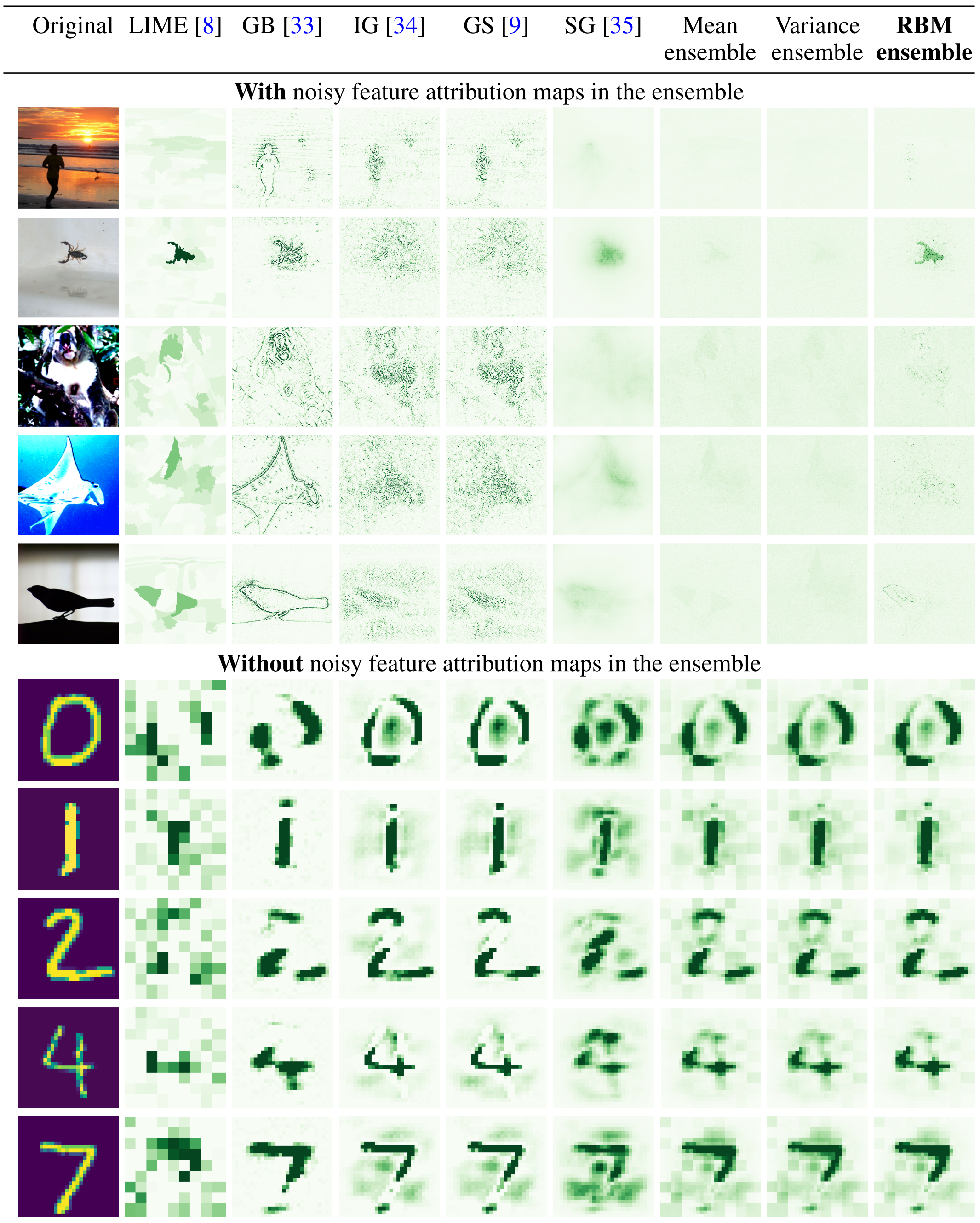

Original
LIME [8]
GB [33]
IG [34]
GS [9]
SG [35]
Mean ensemble
Variance ensemble
RBM ensemble
With noisy feature attribution maps in the ensemble
Without noisy feature attribution maps in the ensemble

Table 3: A visual comparison between base learners and ensemble methods on ImageNet [26] and MNIST [27] datasets.

A Additional Experiments

Table 3 presents extended experimental results for the compression with or without noisy feature attribution maps in the ensemble.

A.2.3 Consistent and Efficient Evaluation Strategy for Attribution Methods

Publication: Published in International Conference on Machine Learning (ICML) 2022.

Contribution: I contributed to the process of idea development, while Tobias Leemann and Yao Rong authored the majority of the paper and implemented the ROAD framework. All co-authors, myself included, actively participated in the revision process to guarantee the quality of the final manuscript

A Consistent and Efficient Evaluation Strategy for Attribution Methods

Yao Rong [*1] **Tobias Leemann** [*1] **Vadim Borisov** [1] **Gjergji Kasneci** [1] **Enkelejda Kasneci** [1]

Abstract

With a variety of local feature attribution methods being proposed in recent years, follow-up work suggested several evaluation strategies. To assess the attribution quality across different attribution techniques, the most popular among these evaluation strategies in the image domain use pixel perturbations. However, recent advances discovered that different evaluation strategies produce conflicting rankings of attribution methods and can be prohibitively expensive to compute. In this work, we present an information-theoretic analysis of evaluation strategies based on pixel perturbations. Our findings reveal that the results are strongly affected by information leakage through the shape of the removed pixels as opposed to their actual values. Using our theoretical insights, we propose a novel evaluation framework termed Remove and Debias (ROAD) which offers two contributions: First, it mitigates the impact of the confounders, which entails higher consistency among evaluation strategies. Second, ROAD does not require the computationally expensive retraining step and saves up to 99 % in computational costs compared to the state-of-the-art. We release our source code at https://github.com/tleemann/road_evaluation.

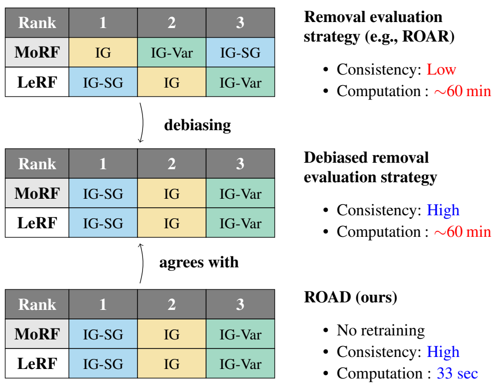


Figure 1. Comparison between previous removal and retraining evaluation strategies (**Top**) and ours (**Bottom**). Previously, rankings of different attribution methods, Integrated Gradients (IG) (Sundararajan et al., 2017) and its two variants SmoothGrad (IG-SG) (Smilkov et al., 2017), SmoothGrad^2 (IG-SQ) (Hooker et al., 2019), are highly inconsistent with respect to hyperparameters such as the removal orders Most Relevant First (MoRF) and Least Relevant First (LeRF). Our ROAD strategy achieves a consistent ranking using only 1% of the previously required resources.

1. Introduction

Explainable Artificial Intelligence (XAI) has become a widely discussed research topic (Adadi & Berrada, 2018). Specifically, feature attribution methods (Springenberg et al., 2015; Ribeiro et al., 2016; Lundberg & Lee, 2017; Sundararajan et al., 2017; Selvaraju et al., 2017) that quantify the importance of input features to a model's decision are widely used. Such local explanations can help to analyze and debug predictive models (Bhatt et al., 2020b; Adebayo et al., 2020), e.g., in the medical domain (Eitel et al., 2019), in recommender systems (Afchar & Hennequin, 2020), and many other applications. With an increasing number of feature attribution methods proposed in the literature, the need for sound strategies to evaluate these methods is also increasing (Nguyen & Martínez, 2020; Hase & Bansal, 2020; Yeh et al., 2019; Hooker et al., 2019).

Evaluation strategies, proposed to compare different attribution methods, commonly follow an ablation approach by perturbing the input features, e.g., image pixels, deemed most or least important. Specifically, perturbing pixels assigned high importance should decrease predictive quality whereas perturbing unimportant pixels, should hardly affect the predictions. These measures aim to capture the *fidelity* of explanations (Tomsett et al., 2020), i.e., how well the explanation genuinely reflects the prediction of the

[*]Equal contribution [1]Department of Computer Science, University of Tübingen, Tübingen, Germany. Correspondence to: Yao Rong <yao.rong@uni-tuebingen.de>, Tobias Leemann <tobias.leemann@uni-tuebingen.de>.

Proceedings of the 39th International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

underlying model. Fidelity based on a single data sample is known as local fidelity, while global fidelity is measured on the whole data set (Tomsett et al., 2020).

The outcome of evaluation strategies is highly sensitive to parameters such as the perturbation function and order. Depending on the order chosen, i.e., *most relevant pixels first* or *least relevant pixels first*, such removal strategies often lead to highly contradictory results. For instance, local attribution methods that seem to perform well in one order may perform rather poorly in the other (Tomsett et al., 2020; Haug et al., 2021; Hooker et al., 2019). This inconsistency makes it hard for researchers to impartially compare between different attribution methods and it is not well understood where the inconsistencies stem from. Moreover, for conducting the global fidelity check, a retraining step is required by some methods (Hooker et al., 2019), which is prohibitively expensive in practice (Tomsett et al., 2020). These two drawbacks and our improvements are illustrated in Figure 1.

In this paper, we aim to overcome these shortcomings and make the evaluation more consistent and efficient. To this end, we propose a new debiased strategy that compensates for confounders causing inconsistencies. Furthermore, we show that in the debiased setting, we can skip the retraining without significant changes in the results. This results in drastic efficiency gains as shown in the lower part of Figure 1. We argue that it is crucial for the community to have sound evaluation strategies that do not suffer from limited accessibility due the required compute capacity. Specifically, we make the following contributions:

- We examine the mechanisms underlying the evaluation strategies based on perturbation by conducting a rigorous information-theoretic analysis, and formally reveal that results can be significantly confounded.
- To compensate for this confounder, we propose the Noisy Linear Imputation strategy and empirically prove its efficiency and effectiveness. The proposed strategy significantly decreases the sensitivity to hyperparameters such as the removal order.
- We generalize our findings to a novel evaluation strategy, ROAD (RemOve And Debias), which can be used to objectively and efficiently evaluate several attribution methods. Compared to previous evaluation strategies requiring retraining, e.g., Remove and Retrain (ROAR) (Hooker et al., 2019), ROAD saves 99 % of the computational costs.

2. Related Work

There is a plethora of works on different explanation techniques (Tjoa & Guan, 2020), especially attribution methods that assign importance scores to each input features. Popular approaches have been proposed by Springenberg et al. (2015); Lapuschkin et al. (2015); Ribeiro et al. (2016); Kasneci & Gottron (2016); Sundararajan et al. (2017); Fong & Vedaldi (2017); Shrikumar et al. (2017); Smilkov et al. (2017); Petsiuk et al. (2018); Adebayo et al. (2018); Chen et al. (2018); Xu et al. (2020); Covert et al. (2021), and many more.

With the growing number of attribution methods, various scholars have presented desiderata that explanations should fulfill (Bhatt et al., 2020a; Nguyen & Martínez, 2020; Fel et al., 2021; Afchar et al., 2021; Nauta et al., 2022). Doshi-Velez & Kim (2017) consider two subcategories in this field, namely *human-grounded* metrics relying on human judgment and *functional-grounded* metrics. The latter do not require a human-generated ground truth that can be hard or even impossible to obtain. Metrics of this type frequently rely on the idea that if the most important part of the image is changed, the output probability of the given black-box model should also change in return. Examples include the Sensitivity-n measure proposed by Ancona et al. (2017) and the infidelity and max-sensitivity metrics by Yeh et al. (2019). Samek et al. (2016) and Petsiuk et al. (2018) also propose to perturb the pixels in the input image according to the importance scores. However, Hooker et al. (2019) show that the perturbation introduces artifacts and results in a distribution shift, putting these no-retraining approaches in question. They propose the Remove and Retrain (ROAR) framework with an extensive model retraining step to adapt to the distribution shift. Therefore, we distinguish between evaluation methods with *retraining* and *no-retraining* approaches. ROAR has been adopted in several recent studies (Hartley et al., 2020; Izzo et al., 2020; Meng et al., 2021; Schramowski et al., 2020; Srinivas & Fleuret, 2019) and variations are being proposed in concurrent work (Shah et al., 2021).

Only few papers have used and compared different evaluation strategies for attribution methods and a sound theoretical explanation for the differences between them is still missing. Sturmfels et al. (2020) assess different baselines for feature attribution applying the Integrated Gradient method (Sundararajan et al., 2017). They also observe that changing the hyperparameter settings can lead to varying results. Haug et al. (2021) draw the same conclusion for attributions on tabular data. Tomsett et al. (2020) compute the consistency among different, no-retraining evaluation strategies and report an alarmingly low agreement. In this work, we conduct a rigorous analysis of reasons for existing inconsistency and provide a solution to reduce it, which is not studied in previous works. Moreover, our solution also reduces high computational costs caused by retraining.

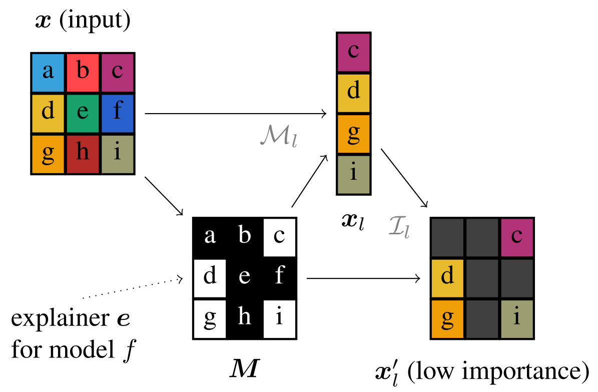


Figure 2. Our analytical model of feature removal evaluation (MoRF order shown): The input image $\boldsymbol{x}$ (9 pixels a–i) is explained by an explanation method that returns a mask $\boldsymbol{M}$ indicating important pixels (black). The remaining, less important pixel values $\boldsymbol{x}_l$ can be extracted from the image using the masking operator $\mathcal{M}_l$ and transformed via the imputation operator $\mathcal{I}_l$ to an imputed variant of the input $\boldsymbol{x}_l'$, which determines the evaluation outcome. This model allows to separate the information in the feature *values* from that contained in the binary mask $\boldsymbol{M}$.

C	Class label random variable
I	Mutual information
$\mathcal{I}$	Imputation operator
$\boldsymbol{M}$	Binary mask in $\{0,1\}^d$
$\mathcal{M}$	Mask selection operator (takes out relevant features)
$\boldsymbol{x}$	Input features in $\mathbb{R}^d$
$\boldsymbol{x}_l$	Low importance features only in $\mathbb{R}^{d-k}$
$\boldsymbol{x}_l'$	Imputed low importance features in $\mathbb{R}^d$

Table 1. Overview of the notation used in this work.

3. Preliminaries

In this section, we formally define the pixel-perturbation strategies considered by the following analysis.

3.1. Retraining Evaluation Strategies

We consider a pixel removal strategy, where pixels are successively replaced by imputed values. Consistent with the literature (Tomsett et al., 2020; Samek et al., 2016), we consider two removal orders: **MoRF** (Most Relevant First) or **LeRF** (Least Relevant First), where the subsequent removal starts with the most important pixels for the former and the least important ones for the latter. We now provide a formal definition of MoRF with retraining, i.e., the ROAR benchmark, that will be used throughout our analysis. We always use the MoRF order in the analysis presented in this paper. However, an analogous analysis of its counterpart LeRF is possible without much additional effort and can be found in the appendix.

To ease our derivations, we describe the procedure by a series of operations that can be analyzed independently. A classifier $f : \mathbb{R}^d \to \{1,\dots,c\}$ maps inputs $\boldsymbol{x} \in \mathbb{R}^d$ to labels $C \in \{1,\dots,c\}$, where c is the number of classes. A feature attribution explanation for the prediction assigns each input dimension an importance value. In the MoRF setting, the features are ordered in a descending order of importance. Subsequently, the k most important features per instance are selected for removal, where $0 \leq k \leq d$ is successively increased during the benchmark. However, for the moment we consider only one fixed value of k. Thus, we can model the explanation $\boldsymbol{e}_k$ as a choice of features via a binary mask $\boldsymbol{M} = \boldsymbol{e}_k(f, \boldsymbol{x}) \in \{0,1\}^d$, with the corresponding value set to one, if the corresponding feature is among the top-k, and to zero otherwise. Furthermore, suppose $\mathcal{M}_l : \{0,1\}^d \times \mathbb{R}^d \to \mathbb{R}^{d-k}$ to be the selection operator for the least important dimensions indicated in the mask and $\boldsymbol{x}_l = \mathcal{M}_l(\boldsymbol{M}, \boldsymbol{x})$ to be a vector containing only the remaining features as shown in Figure 2. We suppose that the features preserve their internal order in $\boldsymbol{x}_l$, i.e., features are ordered ascendingly by their original input indices. This definition allows to separately consider the information flow in the feature *mask* $\boldsymbol{M}$ and that in the feature *values* $\boldsymbol{x}_l$.

The ROAR approach measures the accuracy of a newly trained classifier f' on modified samples $\boldsymbol{x}_l' := \mathcal{I}_l(\boldsymbol{M}, \boldsymbol{x}_l)$, where $\mathcal{I}_l : \{0,1\}^d \times \mathbb{R}^{d-k} \to \mathbb{R}^d$ is an imputation operator that redistributes all inputs in the vector $\boldsymbol{x}_l$ to their original positions and sets the remainder to some filling value. In the special case of zero imputation, $\boldsymbol{x}_l' = \mathcal{I}_l(\boldsymbol{M}, \mathcal{M}_l(\boldsymbol{M}, \boldsymbol{x})) = (1 - \boldsymbol{M}) \odot \boldsymbol{x}$. This means the top-$k$ features are discarded. For a better evaluation result, the accuracy should drop quickly with increasing k, indicating that the most influential features were successfully removed.

3.2. Information Theory

We now briefly revisit the central concepts of information theory that will be handy for our analysis and introduce the notation. The fundamental quantity in information theory is the entropy H of a discrete random variable X with support $\operatorname{supp}\{X\}$,

$$H(X) := -\sum_{x\in\operatorname{supp}\{X\}} P(X = x)\log P(X = x). \quad (1)$$

The entropy corresponds to the information gained through observation of a realization of this variable. If the random variable considered can be easily inferred, we use $p(x)$ as a shorthand for $P(X = x)$. Furthermore, we denote the joint entropy between random variables X and Y by $H(X, Y)$, which is equivalent to the entropy of their joint distribution. In accordance with Cover & Thomas (2006), we always

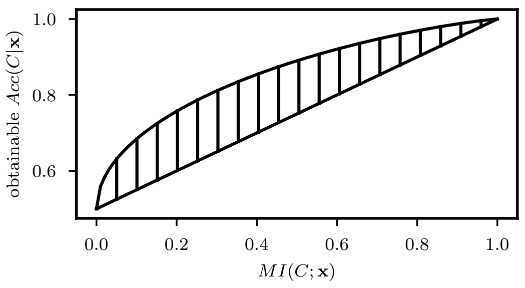


Figure 3. Relation between Mutual Information (MI) and obtainable accuracy for the two-class problem with equal class priors. The knowledge of the MI $I(\boldsymbol{x};C)$ implies strong bounds for the obtainable accuracy. This connection permits to use MI as a surrogate for the obtainable accuracy in the perturbation strategy in our analysis. Figure adapted from Meyen (2016).

separate random variables by comma to denote the joint distribution of multiple of variables.

The conditional entropy $H(X|Y)$ is the expected amount of information left in a variable, given the observation of a condition Y. The most central concept in our analysis will be mutual information (MI), i.e., the amount of information in one random variable shared with another. For example, by $I(\boldsymbol{x};C) := H(C) - H(C|\boldsymbol{x})$, we denote the MI between the complete feature vector and the class variable C. We separate arguments by a semicolon and allow single random variables or sets of random variables as arguments to all the defined quantities. For sets, we always consider the joint distribution of their member variables. Please confer Cover & Thomas (2006) for a more profound introduction. We provide a short overview of our notation in Table 1.

4. Analysis

In this section, we show that the pixel perturbation strategies are susceptible to a previously unknown confounder: The binary mask itself can leak class information that might in not be present in the feature values. After making the connection between the accuracy and mutual information as a theoretical tool in Section 4.1, we formally derive the confounder and identify this leakage on real data in Section 4.2. We subsequently show how to mitigate it through Minimally Revealing Imputation in Section 4.3.

4.1. On the Relation Between Accuracy and Mutual Information

To begin our analysis of the presented strategies and their underlying mechanisms, we first establish the relation between classification accuracy and the mutual information. It is well-known that the classification performance of an optimal classifier in the Bayesian sense (assigning the class with the highest posterior) is dependent on the MI between features and labels (Hellman & Raviv, 1970; Vergara & Estévez, 2014; Meyen, 2016). Nevertheless, the relationship is not a function, but comes in form of upper and lower bounds of the obtainable accuracy. For the simple two-class problem, the bounds are shown in Figure 3 (cf. Appendix A.1 for derivations). They impose strong limits on the optimal classification performance, if the mutual information $I(\boldsymbol{x};C)$ is known.

For the pixel removal strategies that use retraining, this allows us to analyze the frameworks using MI as a surrogate for the attainable accuracy because higher MI almost always leads to higher accuracy. In the MoRF setting with retraining, $I(\boldsymbol{x}'_l;C)$ will play a key role, because it quantifies the information left in the least important features and thus determines obtainable accuracy which is the outcome of the evaluation. Low mutual information $I(\boldsymbol{x}'_l;C)$ results in a sharp drop in accuracy and good benchmarking results:

$$\downarrow I(\boldsymbol{x}'_l;C) \quad \Rightarrow \uparrow \text{MoRF benchmark.}$$

Therefore, in the MoRF setting low mutual information of $\boldsymbol{x}'_l$ and C is desirable[1].

4.2. Class Information Leakage through Masking

We demonstrate that it is easily possible to leak class information only through the mask's shape and to harshly manipulate the evaluation score. Therefore, we start by separating the influence of the mask from that of the feature values. Our derivation relies on the multi-information $I(C;\boldsymbol{x}'_l;\boldsymbol{M})$, which is defined by Vergara & Estévez (2014) as follows:

$$I(C;\boldsymbol{x}'_l;\boldsymbol{M}) = I(C;\boldsymbol{x}'_l|\boldsymbol{M}) - I(C;\boldsymbol{x}'_l) \qquad (2)$$
$$I(C;\boldsymbol{x}'_l;\boldsymbol{M}) = I(C;\boldsymbol{M}|\boldsymbol{x}'_l) - I(C;\boldsymbol{M}). \qquad (3)$$

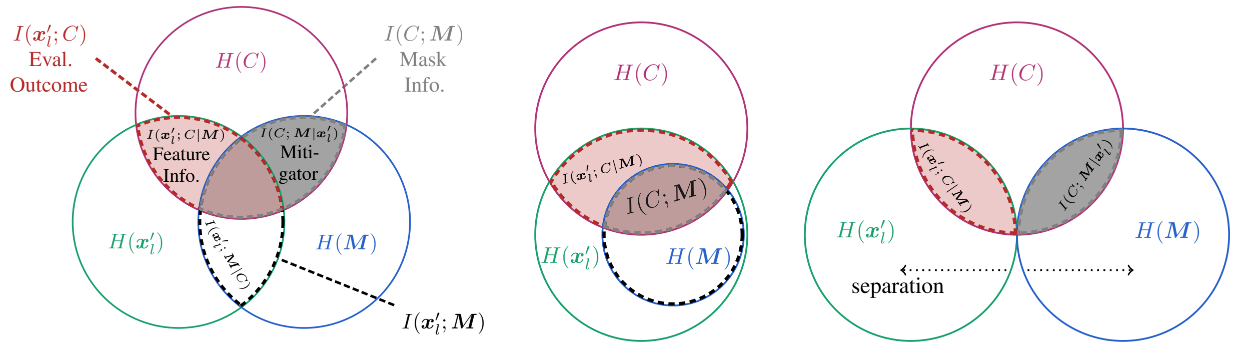
Setting Equation (2) and Equation (3) equal, we arrive at the identity:

$$\underbrace{I(\boldsymbol{x}'_l;C)}_{\text{Eval. Outcome}} = \underbrace{I(C;\boldsymbol{x}'_l|\boldsymbol{M})}_{\text{Feature Info.}} + \underbrace{I(C;\boldsymbol{M})}_{\text{Mask Info.}} - \underbrace{I(C;\boldsymbol{M}|\boldsymbol{x}'_l)}_{\text{Mitigator}}. \qquad (4)$$

The quantities involved are visualized in Figure 4a. The first term "Feature Information" is the class information contained in the features (and not in the mask) that we wish to estimate. The second term "Mask Information" shows that class-discriminative information in the mask can have a high impact on the result. This influence can be compensated by the "Mitigator" term.

Class Information Leakage If the Mask Information term is superior to the Mitigator, $I(C;\boldsymbol{M}) > I(C;\boldsymbol{M}|\boldsymbol{x}'_l)$,

[1]In LeRF, a higher accuracy and thus higher $I(\boldsymbol{x}'_l;C)$ is beneficial


$I(\boldsymbol{x}_l'; C)$
Eval.
Outcome
$H(C)$
$I(C; \boldsymbol{M})$
Mask
Info.
$I(\boldsymbol{x}_l'; C|\boldsymbol{M})$
Feature
Info.
$I(C; \boldsymbol{M}|\boldsymbol{x}_l')$
Miti-
gator
$H(\boldsymbol{x}_l')$
$H(\boldsymbol{M})$
$I(\boldsymbol{x}_l'; \boldsymbol{M}|C)$
$I(\boldsymbol{x}_l'; \boldsymbol{M})$
$H(C)$
$I(\boldsymbol{x}_l'; C|\boldsymbol{M})$
$I(C; \boldsymbol{M})$
$H(\boldsymbol{x}_l')$
$H(\boldsymbol{M})$
$H(C)$
$I(\boldsymbol{x}_l'; C|\boldsymbol{M})$
$I(C; \boldsymbol{M}|\boldsymbol{x}_l')$
$H(\boldsymbol{x}_l')$
$H(\boldsymbol{M})$
separation

(a) General Case (b) Invertible Imputation (c) Minimally Revealing Imputation

Figure 4. The Evaluation Outcome $I(\boldsymbol{x}_l'; C)$ (red area), is confounded by the Mask Information $I(C; \boldsymbol{M})$ (gray area) when there is some overlap (a). Only the Feature Information $I(\boldsymbol{x}_l'; C|\boldsymbol{M})$, the part of the Outcome not overlapping (light red area), should actually be assessed. In the worst case (which we term Invertible Imputation), the Mask Information is entirely contained in the Outcome (b). Separating the information in the imputed image $\boldsymbol{x}_l'$ and the mask $\boldsymbol{M}$ allows to reduce the overlap and the influence (c).

the evaluation outcome is unfairly increased to a value not justified by the selected features. We term this phenomenon *Class Information Leakage*, as some discriminative information is "leaked" through the used binary mask $\boldsymbol{M}$.

The Mitigator can entirely vanish when the mask is perfectly inferable from the imputed image $\boldsymbol{x}_l'$. This results in a non-compensated effect of Class Information Leakage. We define this imputation operation as follows:

Condition 4.1. *Invertible Imputation. Let $\mathcal{I}_l : \{0,1\}^d \times \mathbb{R}^{d-k} \to \mathbb{R}^d$ be the imputation operator that takes the least important features as an input. We suppose that there are inverse functions $\mathcal{I}_{l,M}^{-1}$ and $\mathcal{I}_{l,x}^{-1}$, such that*

$$\boldsymbol{x}_l' = \mathcal{I}_l\left(\boldsymbol{M}, \boldsymbol{x}_l\right) \Leftrightarrow \boldsymbol{M} = \mathcal{I}_{l,M}^{-1}(\boldsymbol{x}_l') \wedge \boldsymbol{x}_l = \mathcal{I}_{l,x}^{-1}(\boldsymbol{x}_l').$$

If, for instance, the pixels removed are set to some reserved value indicating their absence, the imputation operator is invertible, as the mask can be reconstructed. Therefore, $H(\boldsymbol{M}|\boldsymbol{x}_l')=H\left(\mathcal{I}_{l,M}^{-1}(\boldsymbol{x}_l')|\boldsymbol{x}_l'\right)=0$. In this case, also the Mitigator $I\left(C; \boldsymbol{M}|\boldsymbol{x}_l'\right) = 0$, because it is bounded by $0 = H(\boldsymbol{M}|\boldsymbol{x}_l') \geq I\left(C; \boldsymbol{M}|\boldsymbol{x}_l'\right) \geq 0$. The Feature Information term is constrained to be positive. Thus, the Mask Information has a non-negligible impact on the Evaluation Outcome because a higher Mask Information term will always increase it. This case is depicted in Figure 4b.

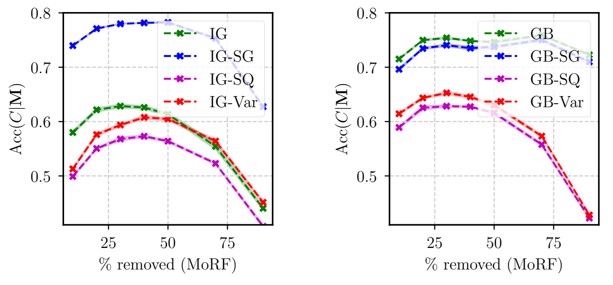
We can create a simple example that shows how evaluation scores are influenced: Imagine a two-class problem that consists of detecting whether an object is located on the left or the right side of an image. A reasonable attribution method masks out pixels on the left or the right depending on the location of the object. In this case, the retraining step can lead to a classifier that infers the class just from the location of the masked out pixels and obtain high accuracy. This explanation map will be rated far worse in MoRF (no accuracy drop) than it might actually be. In the context of amortized explanation methods, a similar finding has been made by Jethani et al. (2021). We theoretically showed that this problem also arises in evaluation strategies and empirically demonstrate that the leakage is significant for popular attribution methods on real data in Section 5.1.

4.3. Reduction of Information Leakage

To tackle this problem, we follow an intuitive approach: If we cannot guarantee that there is no class information contained in the mask itself, we have to stop it from leaking the class information into the imputed images. Therefore, we make sure that the mask used cannot be easily inferred from the imputed image. We would like to set $I(\boldsymbol{x}_l'; \boldsymbol{M}) = 0$, i.e., the mask is independent of the imputed vector allowing to separate the effects as shown in Figure 4c. Unfortunately, this is not possible in general: If both should be dependent on the class label, they will also have to share a minimal amount of information (that regarding the class). However, we can demand conditional independence and make $I(\boldsymbol{x}_l'; \boldsymbol{M})$ as small as possible.

Condition 4.2. *Minimally Revealing Imputation. Let $\mathcal{I}_l : \{0,1\}^d \times \mathbb{R}^{d-k} \to \mathbb{R}^d$ be the infilling operator that takes the least important features as an input. Suppose $\boldsymbol{x}_l'$ and $\boldsymbol{M}$ are independent given the class information $I\left(\boldsymbol{x}_l'; \boldsymbol{M}|C\right) = 0$ and $I\left(\boldsymbol{x}_l'; \boldsymbol{M}\right) \approx 0$.*

In this case, $I(C; \boldsymbol{M}) - I\left(C; \boldsymbol{M}|\boldsymbol{x}_l'\right) = I\left(\boldsymbol{x}_l'; \boldsymbol{M}\right) - I\left(\boldsymbol{x}_l'; \boldsymbol{M}|C\right) \approx 0$, which implies $I(C; \boldsymbol{M}) \approx I\left(C; \boldsymbol{M}|\boldsymbol{x}_l'\right)$ (also cf. Figure 4c), indicating that the Mitigator effectively compensates the Mask Information term.



(a) Integrated Gradients (IG)

(b) Guided Backpropagation (GB)

Figure 5. Accuracy of a trained classifier only using the binary masks $\boldsymbol{M}$ without feature values as input on the CIFAR-10 data set. Binary masks $\boldsymbol{M}$ were computed for different variants of IG and GB. Only the masks contain enough information to reach an accuracy of almost up to 80 % (compared to 85 % with full images) highlighting that the feature values do not play an important role in the evaluation. This underlines the necessity to compensate for this confounder.

5. Debiasing Evaluation Strategies for Local Attribution Methods

With the theoretical analysis in Section 4, we can better understand where the biases come from, and thus mitigate them. Building on the derivations, we now show the strong impact of the Class Information Leakage introduced in Section 4.2 on a real-world data set to highlight the necessity to compensate for this confounder. We explain how we reduce its influence by proposing a novel imputation operator termed *Noisy Linear Imputation*.

5.1. Extent of Class Information Leakage

To empirically confirm our findings, we performed experiments on CIFAR-10 (Krizhevsky et al., 2009). We use the same attribution methods as in Hooker et al. (2019): Integrated Gradients (IG) (Sundararajan et al., 2017) and Guided Backprop (GB) (Springenberg et al., 2015) serve as base explanations, and three ensembling strategies for each are used in addition: SmoothGrad (SG) (Smilkov et al., 2017), SmoothGrad^2 (SQ) (Hooker et al., 2019) and VarGrad (Var) (Adebayo et al., 2018). In total, we consider eight attribution methods and provide details and parameters in the supplementary material.

We empirically show that with fixed value imputation with the global mean, the explanation masks are leaking class information. This takes two steps: (1) We show that the Mask Information $I(C;\boldsymbol{M})$ is extremely high. (2) We verify that the Mitigator is small by testing the *Invertible Imputation* Condition, which implies that class information is leaked into the evaluation outcome through $I(C;\boldsymbol{M})$.

To assess the class information in the mask, we train a ResNet-18 (He et al., 2016) that uses only binary masks $\boldsymbol{M}$ (no pixel values $\boldsymbol{x}_l$) to predict the class. As we discussed previously, the accuracy of a classifier can be used as a surrogate for the calculation of MI, which is prohibitively expensive for high-dimensional data. The curves[2] are shown in Figure 5. Stunningly, the mask alone results in high accuracy curves that reach almost 80 % for IG-SG, only some percent below the accuracy of the classifier on the full inputs. This allows us to conclude that the Mask Information $I\left(C;\boldsymbol{M}\right)$ is almost as high as our Evaluation Outcome $I\left(C;\boldsymbol{x}_l'\right)$.

To show that the Mitigator is almost zero which leads to class information leakage, we test the *Invertible Imputation* condition. Therefore, the inverse function $\mathcal{I}_{l,M}^{-1}$ that predicts the imputation mask from the imputed image is required (having this function, finding $\mathcal{I}_{l,x}^{-1}$ is trivial). For the fixed value imputation, an approximate inverse is simple: Setting all pixels in the mask to 0 if the corresponding image pixel has the filling value (which has to be inferred from the distribution). For a stronger verification, we train an imputation predictor network consisting of three convolutional layers, which predicts for each pixel if it was imputed or original. As Figure 6e (blue curve) shows, the miss-classification rate when using fixed value imputation is almost zero, i.e., the network can easily recognize the pixels that were imputed. According to our analysis, in this setting close to *Invertible Imputation*, the Mitigator will be negligibly small.

This leads us to the conclusion that the mask-related leakage fundamentally influences many previous evaluations using fixed value imputation (Shrikumar et al., 2017; Petsiuk et al., 2018; Hooker et al., 2019) and it is essential to stop the information leakage through the masks.

5.2. Debiasing with Noisy Linear Imputation

To reduce the Class Information Leakage, we propose a better-suited imputation operator $\mathcal{I}_l$ that adheres to the *Minimally Revealing Imputation* condition we derived. The remaining process is left unchanged and stays as depicted in Figure 2. However, we face three requirements: (1) We have to get closer to the theoretical condition of Minimally Revealing Imputation. (2) The imputation strategy needs to be highly efficient, since the imputation module has to be run for each image in the data set. (3) We wish to have as few hyper-parameters as possible (preferably none to rule out another confounding factor).

We devise a new strategy called *Noisy Linear Imputation*, which fulfills the above goals. In this way, our model addresses some of the fundamental problems of existing

[2]Standard Errors are indicated by shaded areas in all figures. However, they are often hardly visible due to their low magnitude.

strategies. Intuitively, we search a way to make more subtle imputations that cannot be easily recognized and result in lower $I(\boldsymbol{x}'_l; \boldsymbol{M})$. To this end, we suppose that each pixel can be approximated by the weighted mean of its neighbors (cf. Figure 6d) as image pixels are highly correlated[3]:

$$\begin{aligned}\boldsymbol{x}_{i,j} &= w_d\left(\boldsymbol{x}_{i,j+1} + \boldsymbol{x}_{i,j-1} + \boldsymbol{x}_{i+1,j} + \boldsymbol{x}_{i-1,j}\right)\\ &\quad + w_i\left(\boldsymbol{x}_{i+1,j+1} + \boldsymbol{x}_{i-1,j+1} + \boldsymbol{x}_{i+1,j-1} + \boldsymbol{x}_{i-1,j-1}\right)\end{aligned}$$

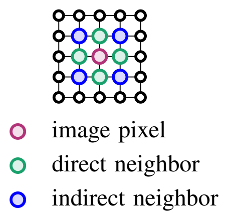
where w_d, w_i are constant coefficients for direct neighbors and indirect, diagonal neighbors. When setting up a single equation for each removed pixel we arrive at an equation system. For known pixels, we directly plug in their values and only consider each removed pixel as an unknown variable. When neighboring pixels are removed, the equations become connected and cannot be solved independently. Nevertheless, the resulting system is sparse and can be efficiently solved, even for a large number of missing pixels. To choose the neighbor weights for the linear interpolation, we draw inspiration from the graph structure (see Figure 6d): Indirect neighbors have distance 2 from the original node in the graph and direct neighbors have distance 1. Hence, we gave the direct neighbors twice the weight of the diagonal ones. Because the weights need to some up to 1 for a weighted interpolation, this leads to $w_d = \frac{1}{6}$ and $w_i = \frac{1}{12}$. We add a small random noise ($\sigma = 0.1$) to the solution to ensure that the linear dependency cannot be learned by the model.

Figure 6 (top) provides an example of an imputed sample. From the imputed version in Figure 6c, inference on the mask is significantly harder than the one imputed with fixed values as in Figure 6b. We again train the imputation predictor for verification and show the results in Figure 6e. We confirm that our strategy lies significantly closer to the optimal, Minimally Revealing Imputation. Admittedly, there are even more sophisticated imputation strategies, for example building on Generative Adversarial Networks (GANs) such as Generative Adversarial Imputation Nets (GAIN) proposed by Yoon et al. (2018). However, our strategy already achieves considerable improvements and is highly efficient, because it does not require training of a GAN model. For completeness, we include additional experiments with GAN imputation in Appendix B.

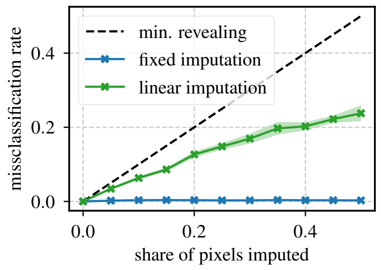
(a) Original

(b) Fixed Imp.

(c) Noisy Lin. Imp.



(d) Graphical model used to derive our imputation



(e) Misclassification rate of the imputation predictor for different shares of randomly imputed pixels (CIFAR-10)

Figure 6. The considered imputation operators. When 50 % of the original image (a) are removed, they can either be imputed by a fixed value (b) or by our proposed Noisy Linear strategy (c,d). Training of an imputation predictor (e) shows that it is much harder to tell which pixels are original and which were imputed when using our proposed imputation model. This is closer to the optimal, minimally revealing imputation (black). Hence, by using imputed samples of this kind, Class Information Leakage is reduced.

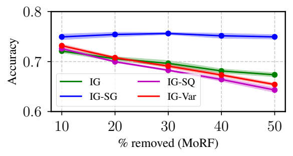
6. Experiments

Having established that our Noisy Linear Imputation fulfills its purpose, in this section, we show that it entails even more benefits in practice. We first highlight how it makes results among different evaluation strategies more consistent in Section 6.1. We then present another considerable advantage in Section 6.2: its agreement with a no-retraining evaluation strategy is sufficiently high, so that the retraining step is no longer required. We name this debiased and no-retraining evaluation framework ROAD (RemOve And Debias). All experiments in this section were conducted on CIFAR-10 using the eight attribution methods mentioned. We also use Food-101 (Bossard et al., 2014), a large-scale dataset of high-resolution images, to validate the generalizability of our method. To this end, we train over 1000 models from scratch on data imputed using the strategies, explanations and removal percentages. Since the results on Food-101 also support the findings from CIFAR-10, we include them in Appendix D.

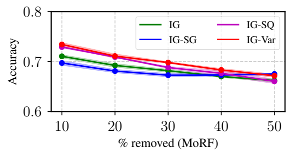
6.1. Consistency under Removal Orders

As we aim for evaluation strategies that are less prone to the hyperparameter setting and allow for a consistent ranking, we study the consistency of evaluation results under the different removal orders MoRF and LeRF. Figure 7 depicts the obtained curves (using "Retrain"). For a clear view, we only show four curves of attribution methods based on IG with retraining and up to 50% pixels are removed. We include the full curves for the IG with its derivatives as well as GB with derivatives in Appendix C. The results using the common fixed value imputation shown in Figure 7a and

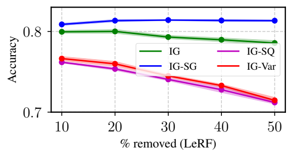
[3]In fact, for direct and indirect neighbors, ρ=0.89 and ρ=0.82 respectively on CIFAR-10



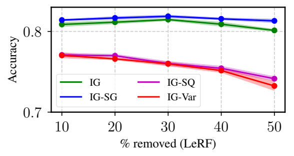
(a) MoRF: Fixed Imp.



(b) MoRF: Noisy Lin. Imp.



(c) LeRF: Fixed Imp.



(d) LeRF: Noisy Lin. Imp.

Figure 7. Consistency comparison using fixed value vs. Noisy Linear Imputation. The higher accuracy is better in LeRF, while the lower is better in MoRF. Comparing (a) and (c), fixed value imputation gives different rankings in MoRF and LeRF orders: IG-SG is the best in LeRF but the worst in MoRF. Comparing (b) and (d), Noisy Linear Imputation changes the outcome considerably and yields a consistent ranking in MoRF and LeRF.

Retrain MoRF vs. LeRF		No-Retrain MoRF vs. LeRF	
fixed	lin	fixed	lin
-0.01±0.01	**0.61**±0.01	0.01±0.00	**0.58**±0.01

Table 2. Spearman rank correlation between evaluation strategies. There is almost no agreement between MoRF and LeRF when using fixed imputation (as in previous works). When using our imputation ("lin"), consistency across MoRF and LeRF orders increases drastically.

MoRF Retain vs. No-Retr.		LeRF Retain vs. No-Retr.	
fixed	lin	fixed	lin
0.15±0.01	**0.84**±0.01	0.09±0.01	**0.94**±0.01

Table 3. Spearman rank correlation between evaluation with and without retraining. Our Noisy Linear Imputation ("lin") also results only in marginal differences between "Retrain" and "No-Retrain". We conclude that the retraining step is no longer necessary.

Figure 7c. The results with our Noisy Linear Imputation are shown in Figure 7b and Figure 7d. In MoRF, a sharp drop in the beginning indicates a better attribution method, while a slight drop is desirable in LeRF. Hence, using fixed imputation, the ranking in MoRF is IG, IG-Var, IG-SQ, IG-SG, whereas the ranking in LeRF is IG-SG, IG, IG-SQ, and IG-Var. We see, for instance, that IG-SG is the worst in MoRF and the best in LeRF. When using the Noisy Linear Imputation, the inconsistency vanishes. The ranking in MoRF is: IG-SG, IG, IG-SQ, and IG-Var, which is the same as in LeRF.

We quantitatively compute the consistency among all eight attribution methods with and without retraining. Concretely, we compute the ranks (from 1=best to 8=worst) of our explanation methods for each percentage of perturbed pixels. We then calculate the Spearman Rank correlation between different evaluation strategies. As shown in Table 2, the correlation score of the fixed value imputation is -0.01 when using retraining and 0.01 when no retraining is applied. This indicates no consistency in the rankings. When we deploy our Noisy Linear Imputation, the results change drastically: The correlation score is improved to 0.61 and 0.58 with and without retraining, respectively. This might imply that the information leakage is responsible for a major share of the inconsistency.

6.2. Efficiency

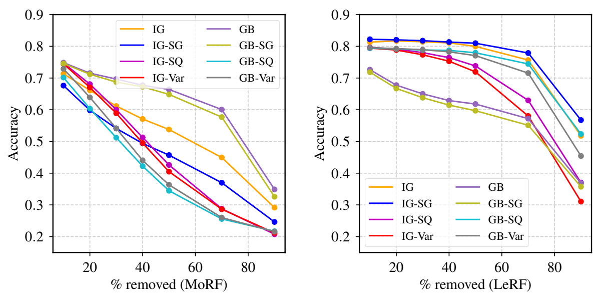
When we apply our Noisy Linear Imputation, we additionally reduce the difference between evaluation with and without retraining. This can be attributed to the reduced distribution shift incurred when using an almost *Minimally Revealing Imputation*. If all pixels were perfectly imputed, the resulting image would not be out-of-distribution. Since we are interested in the rankings of attribution methods, we again compute Spearman correlation between the rankings obtained with and without retraining and show it in Table 3. The order remains almost always intact between the "Retrain" with Noisy Linear Imputation and the "No-Retrain" variant with Noisy Linear Imputation resulting in a rank correlation of 0.84 in using MoRF and 0.94 in LeRF. This leads us to the conclusion that "No-Retrain" and "Retrain" end up with a highly similar ranking when using Noisy Linear Imputation. Thus, we conclude that the retraining step is not longer justified and can be skipped without significant distortion of the results. Qualitative results are shown in Appendix C.3, cf. Figure 17 (CIFAR-10) and Figure 23 (Food-101).

These results allow us to introduce a novel evaluation framework. We refer to the removal with Noisy Linear Imputation and no retraining as ROAD – Remove and Debias. We showed that ROAD is highly consistent with the compensated results of the ROAR, but comes at an enormous advantage: The retraining step is no longer required. This permits to save a vast amount of computation time. In our experiments, evaluation using the ROAD took only $0.7\,\%$ of the resources required for ROAR, as given by the runtimes in Table 4 obtained on the same hardware (single Nvidia GTX 2080Ti and 8 Cores).

In the end, we illustrate the evaluation results using ROAD among all eight attribution methods in MoRF and LeRF in Figure 8. In MoRF, the best ones are IG-SG, GB-SQ, GB-Var and IG, which have lower accuracies in the beginning, whereas they have higher accuracies in LeRF.

Strategy	Retrain		No-Retrain	
	fixed†	lin	fixed	lin*
Time	3903±117 s	4686±2 s	18.0±0.1 s	33.3±0.1 s
Relative	100 %	120 %	0.5 %	0.9 %

Table 4. Mean runtime (5 runs) for evaluating a single explanation method (IG). † refers to ROAR, and $\star$ to our ROAD.



(a) MoRF (b) LeRF

Figure 8. Evaluation results in MoRF (a) and LeRF (b) using our ROAD framework.

GB and GB-Var both perform badly in MoRF and LeRF. We see that some inconsistencies still remain, which cannot be compensated by the current imputation. However, the evaluation strategies might also consider different characteristics of an attribution method (e.g., one might be particularly good at identifying irrelevant pixels), which is why perfect agreement might not even be desirable.

7. Conclusion and Outlook

We introduced ROAD, an evaluation approach for measuring global fidelity among attribution explanations. ROAD comes with two key advantages over existing methods: (1) it is highly efficient, e.g., permitting a 99% runtime reduction w.r.t. ROAR, and (2) it circumvents the Class Information Leakage issue, which was thoroughly analyzed in this work. We believe the ROAD framework will be beneficial to the research community because it unifies several methods and is more consistent under varying removal orders. Moreover, it is broadly accessible due to its low resource requirements. ROAD is open-source[4], and can be readily implemented in practical use-cases. Going forward, we plan to investigate more sophisticated imputation models in ROAD as well as other evaluation metrics besides fidelity.

Acknowledgements

We acknowledge the support by the Cluster of Excellence - Machine Learning: New Perspectives for Science, EXC number 2064/1 - Project number 390727645, and the support of the Training Center for Machine Learning (TCML) Tübingen, funded by the German Federal Ministry of Education and Research (BMBF) with grant number 01IS17054, which provided substantial resources for running our large-scale Food-101 experiment.

[4]An official implementation is also included in the Quantus framework (Hedström et al., 2022)

References

Adadi, A. and Berrada, M. Peeking inside the black-box: a survey on explainable artificial intelligence (xai). *IEEE access*, 6:52138–52160, 2018.

Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., and Kim, B. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Adebayo, J., Muelly, M., Liccardi, I., and Kim, B. Debugging tests for model explanations. *arXiv preprint arXiv:2011.05429*, 2020.

Afchar, D. and Hennequin, R. Making neural networks interpretable with attribution: application to implicit signals prediction. In *Fourteenth ACM Conference on Recommender Systems*, pp. 220–229, 2020.

Afchar, D., Guigue, V., and Hennequin, R. Towards rigorous interpretations: a formalisation of feature attribution. In *International Conference on Machine Learning*, pp. 76–86. PMLR, 2021.

Ancona, M., Ceolini, E., Öztireli, C., and Gross, M. Towards better understanding of gradient-based attribution methods for deep neural networks. *arXiv preprint arXiv:1711.06104*, 2017.

Bhatt, U., Weller, A., and Moura, J. M. Evaluating and aggregating feature-based model explanations. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pp. 3016 –3022, 2020a.

Bhatt, U., Xiang, A., Sharma, S., Weller, A., Taly, A., Jia, Y., Ghosh, J., Puri, R., Moura, J. M., and Eckersley, P. Explainable machine learning in deployment. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pp. 648–657, 2020b.

Bossard, L., Guillaumin, M., and Gool, L. V. Food-101–mining discriminative components with random forests. In *European conference on computer vision*, pp. 446–461. Springer, 2014.

Chen, J., Song, L., Wainwright, M. J., and Jordan, M. I. L-shapley and c-shapley: Efficient model interpretation for structured data. In *International Conference on Learning Representations*, 2018.

Cover, T. M. and Thomas, J. A. *Elements of Information Theory*. John Wiley and Sons, 2006. doi: 10.1002/047174882X.

Covert, I., Lundberg, S., and Lee, S.-I. Explaining by removing: A unified framework for model explanation. *Journal of Machine Learning Research*, 22(209):1–90, 2021.

Doshi-Velez, F. and Kim, B. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

Eitel, F., Ritter, K., Alzheimer's Disease Neuroimaging Initiative (ADNI), et al. Testing the robustness of attribution methods for convolutional neural networks in mri-based alzheimer's disease classification. In *Interpretability of Machine Intelligence in Medical Image Computing and Multimodal Learning for Clinical Decision Support*, pp. 3–11. Springer, 2019.

Fel, T., Vigouroux, D., Cadène, R., and Serre, T. How good is your explanation? algorithmic stability measures to assess the qualityof explanations for deep neural networks. 2021.

Fong, R. C. and Vedaldi, A. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE international conference on computer vision*, pp. 3429–3437, 2017.

Hartley, T., Sidorov, K., Willis, C., and Marshall, D. Explaining failure: Investigation of surprise and expectation in cnns. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 12–13, 2020.

Hase, P. and Bansal, M. Evaluating explainable AI: Which algorithmic explanations help users predict model behavior? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 5540–5552, 2020.

Haug, J., Zürn, S., El-Jiz, P., and Kasneci, G. On baselines for local feature attributions. *AAAI Workshop on Explainable Agency in AI Workshop*, 2021.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Hedström, A., Weber, L., Bareeva, D., Motzkus, F., Samek, W., Lapuschkin, S., and Höhne, M. M.-C. Quantus: an explainable AI toolkit for responsible evaluation of neural network explanations. *arXiv preprint arXiv:2202.06861*, 2022.

Hellman, M. E. and Raviv, J. Probability of Error, Equivocation, and the Chernoff Bound. *IEEE Transactions on Information Theory*, 16(4):368–372, 1970.

Hooker, S., Erhan, D., Kindermans, P. J., and Kim, B. A benchmark for interpretability methods in deep neural networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Izzo, C., Lipani, A., Okhrati, R., and Medda, F. A baseline for shapely values in mlps: from missingness to neutrality. *arXiv preprint arXiv:2006.04896*, 2020.

Jethani, N., Sudarshan, M., Aphinyanaphongs, Y., and Ranganath, R. Have we learned to explain?: How interpretability methods can learn to encode predictions in their interpretations. In *International Conference on Artificial Intelligence and Statistics*, pp. 1459–1467. PMLR, 2021.

Kachuee, M., Karkkainen, K., Goldstein, O., Darabi, S., and Sarrafzadeh, M. Generative imputation and stochastic prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

Kasneci, G. and Gottron, T. Licon: A linear weighting scheme for the contribution ofinput variables in deep artificial neural networks. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pp. 45–54, 2016.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Lapuschkin, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., and Samek, W. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.

Lundberg, S. M. and Lee, S.-I. A unified approach to interpreting model predictions. In *Proceedings of the 31st international conference on neural information processing systems*, pp. 4768–4777, 2017.

Meng, C., Trinh, L., Xu, N., and Liu, Y. Mimic-if: Interpretability and fairness evaluation of deep learning models on mimic-iv dataset. *arXiv preprint arXiv:2102.06761*, 2021.

Meyen, S. Relation between classification accuracy and mutual information in equally weighted classification task. Master's thesis, University of Hamburg, 2016. URL https://osf.io/zru7b/.

Nauta, M., Trienes, J., Pathak, S., Nguyen, E., Peters, M., Schmitt, Y., Schlötterer, J., van Keulen, M., and Seifert, C. From anecdotal evidence to quantitative evaluation methods: A systematic review on evaluating explainable ai. *arXiv preprint arXiv:2201.08164*, 2022.

Nguyen, A. P. and Martínez, M. R. On quantitative aspects of model interpretability. *arXiv preprint arXiv:2007.07584*, 2020.

Petsiuk, V., Das, A., and Saenko, K. Rise: Randomized input sampling for explanation of black-box models. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2018.

Ribeiro, M. T., Singh, S., and Guestrin, C. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144, 2016.

Samek, W., Binder, A., Montavon, G., Lapuschkin, S., and Müller, K.-R. Evaluating the visualization of what a deep neural network has learned. *IEEE transactions on neural networks and learning systems*, 28(11):2660–2673, 2016.

Schramowski, P., Stammer, W., Teso, S., Brugger, A., Herbert, F., Shao, X., Luigs, H.-G., Mahlein, A.-K., and Kersting, K. Making deep neural networks right for the right scientific reasons by interacting with their explanations. *Nature Machine Intelligence*, 2(8):476–486, 2020.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

Shah, H., Jain, P., and Netrapalli, P. Do input gradients highlight discriminative features?, 2021.

Shrikumar, A., Greenside, P., and Kundaje, A. Learning important features through propagating activation differences. In *International Conference on Machine Learning*, pp. 3145–3153. PMLR, 2017.

Smilkov, D., Thorat, N., Kim, B., Viégas, F., and Wattenberg, M. Smoothgrad: removing noise by adding noise. In *Workshop on Visualization for Deep Learning, ICML*, 2017.

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. Striving for simplicity: The all convolutional net. In *ICLR (workshop track)*, 2015.

Srinivas, S. and Fleuret, F. Full-gradient representation for neural network visualization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Sturmfels, P., Lundberg, S., and Lee, S.-I. Visualizing the impact of feature attribution baselines. *Distill*, 5(1):e22, 2020.

Sundararajan, M., Taly, A., and Yan, Q. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pp. 3319–3328. PMLR, 2017.

Sutskever, I., Martens, J., Dahl, G., and Hinton, G. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pp. 1139–1147. PMLR, 2013.

Tjoa, E. and Guan, C. A survey on explainable artificial intelligence (xai): Toward medical xai. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

Tomsett, R., Harborne, D., Chakraborty, S., Gurram, P., and Preece, A. Sanity checks for saliency metrics. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 6021–6029, 2020.

Vergara, J. R. and Estévez, P. A. A review of feature selection methods based on mutual information. *Neural Computing and Applications*, 2014.

Xu, S., Venugopalan, S., and Sundararajan, M. Attribution in scale and space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9680–9689, 2020.

Yeh, C.-K., Hsieh, C.-Y., Suggala, A., Inouye, D. I., and Ravikumar, P. K. On the (in) fidelity and sensitivity of explanations. *Advances in Neural Information Processing Systems*, 32:10967–10978, 2019.

Yoon, J., Jordon, J., and Schaar, M. Gain: Missing data imputation using generative adversarial nets. In *International Conference on Machine Learning*, pp. 5689–5698. PMLR, 2018.

A. Additional Theory

A.1. Formulation of the MI Bounds for the Binary Case

As we discussed in our main paper, the relationship between Mutual Information (MI) and accuracy is not a function, but comes in form of upper and lower bounds of the obtainable accuracy. If, for example, the binary classification case with equal class priors $p(C=0)=p(C=1)=\frac{1}{2}$ is considered, the following bounds can be derived (Hellman & Raviv, 1970; Meyen, 2016):

$$\frac{I(\boldsymbol{x};C)+1}{2} \leq \mathrm{Acc}(C|\boldsymbol{x}) \leq H_2^{-1}(1-I(\boldsymbol{x};C)), \tag{5}$$

where $H_2^{-1}:[0,1]\to\left[\frac{1}{2},1\right]$ is the inverse of the binary entropy with support $\left[\frac{1}{2},1\right]$. For completeness, we restate the proof of this upper bound in Appendix A.2.

A.2. Reproduction of the proof of the relation between mutual and accuracy in the binary case

In this section, we reproduce the proofs for the upper and lower bounds of bayesian classifier accuracy given a certain amount of mutual information from the master's thesis by (Meyen, 2016) for completeness. The upper bound given there is tighter than the bounds present in the literature.
We consider the following setting (C, $\boldsymbol{x}$ are random variables):

- binary classification problem, $C\in\Omega_C=\{0,1\}$
- equal class priors $P(C=0)=\frac{1}{2}, P(C=1)=\frac{1}{2}$
- discrete features $\boldsymbol{x}$ (which can be the product of multiple random variables)
- support set $\Omega_x=\mathrm{supp}\,\{\boldsymbol{x}\}$ of *countable* size

We first prove the following Lemma:

Lemma A.1. *Let the assumptions stated above be true. Then, the mutual information is the weighted mean of a function of the conditional accuracies* $\mathrm{Acc}(C|s)$*, where* $s\in\Omega_x$*:*

$$I\left(C;\boldsymbol{x}\right)=\sum_{s\in\Omega_S}p(s)\left(1-H_2\left[\mathrm{Acc}(C|s)\right]\right)$$

In this formulation, $p(s)$ is a shorthand for $P(\boldsymbol{x}=s)$ and $H_2(p):=-p\log p-(1-p)\log(1-p)$ is the entropy for a binary random variable.
Proof.

$$\begin{aligned}
I\left(C;\boldsymbol{x}\right) &= H(C)-H(C|\boldsymbol{x}) && (6)\\
&= \sum_{c\in\Omega_C}p(c)\log\frac{1}{p(c)}-\sum_{s\in\Omega_S}p(s)\sum_{c\in\Omega_C}p(c|s)\log\frac{1}{p(c|s)} && (7)\\
&= \sum_{s\in\Omega_x}p(s)\left[\sum_{c\in\Omega_C}p(c)\log\frac{1}{p(c)}-\sum_{c\in\Omega_C}p(c|s)\log\frac{1}{p(c|s)}\right] && (8)\\
&= \sum_{s\in\Omega_x}p(s)\left[H(C)-H(C|s)\right] && (9)
\end{aligned}$$

In our consideration, $\Omega_C=\{0,1\}$ and $P(C=0)=\frac{1}{2}, P(C=1)=\frac{1}{2}$, so $H(C)=1$. Additionally, the bayesian classifier rule yields

$$acc(C|s)=\begin{cases} P(C=0|s), & \text{for } P(C=1|s)\leq 0.5\\ P(C=1|s), & \text{for } P(C=1|s)>0.5\end{cases} \tag{10}$$

and

$$\begin{aligned} H(C|s) &= -P(C=0|s)\log P(C=0|s) - P(C=1|s)\log P(C=1|s) && (11)\\ &= H_2(P(C=0|s)) = H_2(P(C=1|s)) && (12)\\ &= H_2(acc(C|s)) && (13)\end{aligned}$$

Plugging in the results $H(C) = 1$ and $H(C|s) = H_2(\mathrm{Acc}(C|s))$, we obtain the proposed lemma. □

For the derivation of upper and lower bounds, Jenssen's inequality is used. $1 - H_2(\cdot)$ is a convex function and the $\{p(s)\}_{s\in\Omega_x}$ are convex multipliers, i.e., they are non-negative and sum up to one. Then,

$$\begin{aligned} 1 - H_2(\mathrm{Acc}(C|\boldsymbol{x})) &= 1 - H_2\left(\sum_{s\in\Omega_x} p(s)\,\mathrm{Acc}(C|s)\right) && (14)\\ &\le \sum_{s\in\Omega_x} p(s)\left[1 - H_2(\mathrm{Acc}(C|s))\right] = I(\boldsymbol{x};C) && (15)\end{aligned}$$

We can restate this equation in terms of accuracy.

$$H_2(\mathrm{Acc}(C|\boldsymbol{x})) \ge 1 - I(C;\boldsymbol{x}) \tag{16}$$

Using that $H_2(\cdot)$ is decreasing monotonically on the interval $\left[\frac{1}{2}, 1\right]$, so its inverse H_2^{-1} exists, and that $\mathrm{Acc}(C|s) \ge 0.5$:

$$\mathrm{Acc}(C|\boldsymbol{x}) \le H_2^{-1}(1 - I(C;\boldsymbol{x}))\,. \tag{17}$$

The inequality sign is flipped again, due to the inverse being monotonically decreasing. Note that the bounds derived for the special case are much tighter than the general ones provided by Vergara & Estévez (2014) and Cover & Thomas (2006, Chapter 2.10), that are not of any use, because they are even less strict than the trivial bound $\mathrm{Acc}(C|\boldsymbol{x}) \le 1$, for the simple case considered here.

For the lower bound, we refer the reader to Hellman & Raviv (1970, eqn. 18), where the term I corresponds to $H(C|\boldsymbol{x}) = H(C) - I(C;\boldsymbol{x})$ in our notation. Rewriting the result from Hellman & Raviv (1970) in our notation, we obtain

$$1 - \mathrm{Acc}(C|\boldsymbol{x}) \le \frac{H(C) - I(C;\boldsymbol{x})}{2}. \tag{18}$$

Using $H(C) = 1$ and rearranging yields

$$1 - \mathrm{Acc}(C|\boldsymbol{x}) \le \frac{1 - I(C;\boldsymbol{x})}{2} \tag{19}$$

and

$$\mathrm{Acc}(C|\boldsymbol{x}) \ge \frac{I(C;\boldsymbol{x}) + 1}{2}. \tag{20}$$

□

A.3. Analysis of the LeRF Ordering

In this section, we analyze the masking impact for the case of the Least Relevant First (LeRF) ordering. We first provide a definition for the operators involved as we did for the Most Relevant First (MoRF) case. In the LeRF setting, the k least important important features per instance are removed. We model the explanation as a choice of features via a binary mask $\boldsymbol{M} = \boldsymbol{e}(f, \boldsymbol{x}) \in \{0,1\}^d$, with the corresponding value set to one, if the corresponding feature is among the top-k, and to zero otherwise. Furthermore, suppose $\mathcal{M}_h : \{0,1\}^d \times \mathbb{R}^d \to \mathbb{R}^k$ to be the selection operator for the <u>**h**</u>ighly important dimensions indicated in the mask and $\boldsymbol{x}_h = \mathcal{M}_h(\boldsymbol{M}, \boldsymbol{x})$ to be a vector containing only the remaining, highly important features as shown in Figure 9. We suppose that the features preserve their internal order in $\boldsymbol{x}_h$, i.e., features are ordered ascendingly by their original input indices.

The LeRF approach with retraining (also called "Keep and Retrain", KAR, by Hooker et al. (2019)) measures the accuracy of a newly trained classifier f' on modified samples $\boldsymbol{x}'_h := \mathcal{I}_h(\boldsymbol{M}, \boldsymbol{x_h})$, where $\mathcal{I}_h : \{0,1\}^d \times \mathbb{R}^k \to \mathbb{R}^d$ is an imputation

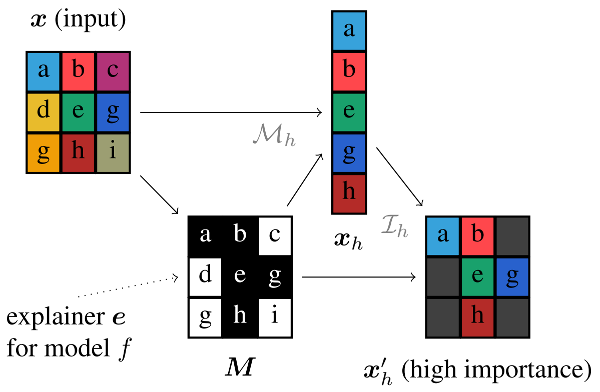


Figure 9. Analogous analytical model of feature removal in the opposite order (LeRF): The input image $\boldsymbol{x}$ is explained by an explanation method that returns a mask $\boldsymbol{M}$ indicating important pixels. The remaining, highly important pixels can be extracted from the image using the masking operator $\mathcal{M}_h$ and transformed to a modified variant of the input $\boldsymbol{x}'_h$ via the imputation operator $\mathcal{I}_h$.

operator that redistributes all inputs in the vector $\boldsymbol{x}_h$ to their original positions and sets the remainder to some filling value. This means only the top-k features are kept. For a better evaluation result, the accuracy should increase quickly with increasing k, indicating the most influential features are present. Accuracy should not increase much for the high values of k, because inserting the low importance features should not have a large effect (equivalently, this means it should not drop much when the least important features are removed). Overall, higher accuracies indicate better attributions in the LeRF setting.

For the LeRF benchmark, the quantity of interest in our analysis will be $I(\boldsymbol{x}'_h; C)$, the class information contained in the filled-in version of the selected high important features. We want to maximize $I(\boldsymbol{x}'_h; C)$ to obtain a good score,

$$\uparrow I(\boldsymbol{x}'_h; C) \quad \Rightarrow \uparrow \text{LeRF benchmark.}$$

As before, we can apply the following, general identity:

$$\underbrace{I(\boldsymbol{x}'_h; C)}_{\text{Evaluation Outcome}} = \underbrace{I(C; \boldsymbol{x}'_h | \boldsymbol{M})}_{\text{Feature Info.}} + \underbrace{I(C; \boldsymbol{M})}_{\text{Mask Info.}} - \underbrace{I(C; \boldsymbol{M} | \boldsymbol{x}'_h)}_{\text{Mitigator}}. \tag{21}$$

The interpretation of the terms is analogous to that in our main paper.

Class-Leaking Explanation Map For the case of the class-leaking map, we again require the imputation operator to be invertible:

Example A.2. *Invertible Imputation. Let $\mathcal{I}_h : \{0,1\}^d \times \mathbb{R}^k \to \mathbb{R}^d$ be the imputation operator that takes the highly important features as an input. We suppose that there are inverse functions $\mathcal{I}^{-1}_{h,M}$ and $\mathcal{I}^{-1}_{h,x}$, such that*

$$\boldsymbol{x}'_h = \mathcal{I}_h(\boldsymbol{M}, \boldsymbol{x}_h) \Leftrightarrow \boldsymbol{M} = \mathcal{I}^{-1}_{h,M}(\boldsymbol{x}'_h) \wedge \boldsymbol{x}_h = \mathcal{I}^{-1}_{h,x}(\boldsymbol{x}'_h).$$

If, for instance, the pixels removed are set to some reserved value indicating their absence, the infilling operator is invertible. In this case, also the Mitigator $I(C; \boldsymbol{M}|\boldsymbol{x}'_h) = 0$ (see Section 4.3 for details). The "Feature Info" term is constrained to be positive. Thus, the Mask Information has a non-negligible impact on the Evaluation Goal, because a higher Mask term will always increase it.

We can create a another example of a spurious explanation map that shows how evaluation scores are influenced even worse for LeRF: Suppose an explanation map that starts masking out pixels at the top for class zero and at the bottom for class one. Thus, a retrained model will be able to infer the category just from the shape of the masked pixels and obtain the best possible accuracy and thus score in the LeRF setting. However, it does not provide a reasonable attribution for the importance of the features.

B. GAN Imputation

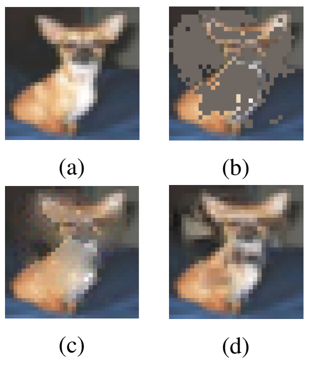
We also use Generative Adversarial Imputation Nets (GAIN) proposed by Yoon et al. (2018) as an imputation operator. We first train a GAIN model on CIFAR-10. To find the best-performing setup, we run a hyperparameter selection for the GAIN model. We keep all the default parameters identified by Kachuee et al. (2020), but search for the value of `alpha` (α), which can be seen as a weight factor for the reconstruction loss of the non-imputed pixels in the GAN, and the `hint_rate` (hr) parameter, which provides the Discriminator with hints to balance the difficulty of the tasks. We train the models for 100 epochs which resulted in converged MSEs and Frechet Inception Distances (FIDs). We use MSE to the original pixels to assess the generative quality of the model. Kachuee et al. (2020) reported low values for both these parameters to perform well, but did not provide the exact values. We extended their value ranges to $\alpha = 100$ and performed and exhaustive search. The results for the GAIN models on CIFAR-10 can be seen in Table 5. For the experiments we used the best setup with $\alpha = 100$ and $hr = 0.01$.

	α=0.1	α=1	α=10	α=100
hr=0.01	0.0131	0.0164	0.0090	**0.0085**
hr=0.1	0.0113	0.0133	0.0131	0.0101
hr=0.3	0.0172	0.0183	0.0151	0.0127
hr=0.9	0.0303	0.0484	0.0379	0.0088

Table 5. Mean-Squared-Errors for GAIN on CIFAR-10 using different hyperparameter choices.

In Figure 10, we demonstrate imputation results using three operators for one image (a) from CIFAR-10. Compared to the fixed value imputation (b) and noisy linear imputation (c), GAN imputation (d) yields most natural imputed image. Although it cannot perfectly reconstruct the original image, for example the background is noisy and the body color is different from the original one, it is not easy to deduce the mask from (d). A trained imputation predictor also verifies that GAN imputation is closest to the optimal condition, Minimally Revealing Imputation.

However, there are drawbacks of the GAN imputation. It may introduce some new "features" that do not exist in the original sample. For instance the dog in (d) has new patterns on its body. Moreover, it does not give very good results when too many pixels are removed (cf. Figure 12). The GAIN training again requires tuning hyperparameter settings and is highly expensive. Therefore, this model does not allow for the desired improvements (few hyperparameters, efficiency). Compared to GAN, our Noisy Linear imputation does not have these drawbacks. Considering all these factors, we recommend to use Noisy Linear Imputation in the evaluation framework.



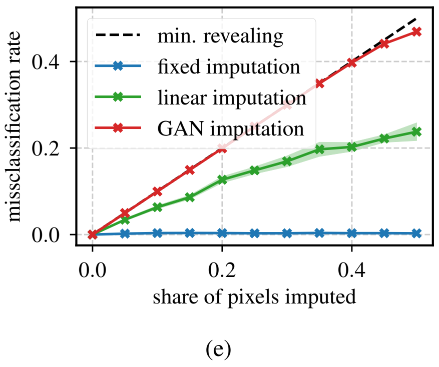


Figure 10. The considered imputation operators. When 30 % of the original image (a) are removed, they can either be completed by a fixed value (b) or by our proposed Noisy Linear imputation (c) or GAN imputation (d). Training of an imputation predictor (e) shows that it is much harder to tell which pixels are original and which were imputed when using our proposed imputation models, which is closer to the theoretical optimum (black). Hence, Class Information Leakage is reduced by our imputation methods.

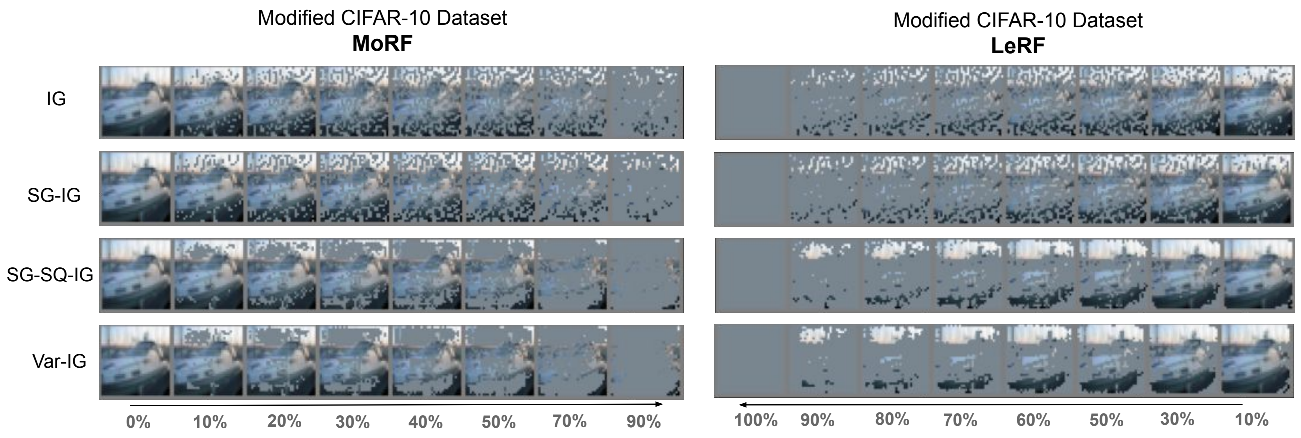


Figure 11. Illustration of modified data set in MoRF/LeRF and fixed value imputation settings. **Left**: Modifications in the MoRF framework. **Right**: Modifications in the LeRF framework. **Top to Bottom**: Modifications using Integrated Gradient (IG) (Sundararajan et al., 2017) and three ensemble variants of IG: SmoothGrad (SG-IG) (Smilkov et al., 2017), SmoothGrad^2 (SG-SQ-IG) (Hooker et al., 2019), and VarGrad (Var-IG) (Adebayo et al., 2018). The percentage of pixels that are removed or kept is given at the bottom.

C. Additional Experiments on CIFAR-10

C.1. Implementation Details

In this section, we report implementation details on CIFAR-10 as well as additional results for comparison between fixed value imputation and our *Noisy Linear Imputation*. We also include GAN imputation results. In Figure 12, an overview of using three different imputations with different perturbation percentages are illustrated.

We train a vanilla ResNet-18 (He et al., 2016) on CIFAR-10 and compute different explanations using the trained model. The model is trained with the initial learning rate of 0.01 and the SGD optimizer (Sutskever et al., 2013). We decrease the learning rate by factor 0.1 after 25 and train the model for 40 epochs on one GPU. The trained model achieves a test set accuracy of $84.5\,\%$ (comparable to the model in (Tomsett et al., 2020)). For attributions, we use the same settings as in (Hooker et al., 2019): As base explanations we implement Integrated Gradient (IG) (Sundararajan et al., 2017) and Guided Backprop (GB) (Springenberg et al., 2015). Additionally, we use three ensembling strategies for each: SmoothGrad (SG) (Smilkov et al., 2017), SmoothGrad^2 (SG-SQ) (Hooker et al., 2019) and VarGrad (Var) (Adebayo et al., 2018). For each explanation method, we modify the data set using the fraction of pixels $\eta = [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.7, 0.9]$. Figure 11 illustrates the modified images by using four different explanations in the GB-family within MoRF and LeRF orders (fixed mean value imputation is used).

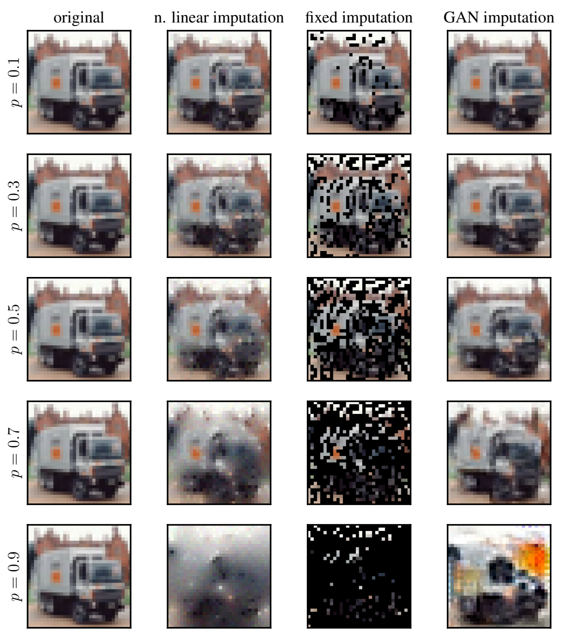
We use $N = 5$ **runs** and report averaged results for all CIFAR-10 experiments in our paper and indicate the standard errors (which are very small) as an area behind our plots. In Table 6 and Table 7, we show the mean accuracy and its standard deviation at each the fraction of pixels η for IG-SG and GB-SG explanations. For other explanations we used, the standard deviation at each η in the magnitude of below one percent as well. Mean runtimes (average over 5 runs) for evaluating one explanation method (IG) using all three imputation methods are listed in Table 8.

C.2. Correlation Analysis

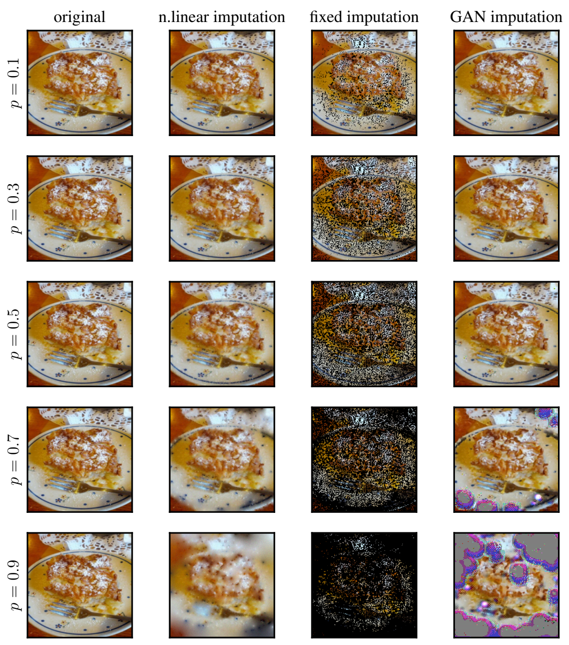
In Table 9, we show a full view of the Spearman Correlation of rankings between all twelve different evaluation strategies ("Retrain"/"No-Retrain", MoRF/LeRF, and fixed value/Noisy Linear/GAN imputation) used in this paper. In this work, our primary focus was on consistency between the respective Retraining/No-Retraining Methods and the consistency between MoRF/LeRF and we mark the results used in the main paper in bold.

C.3. Extended Figures

In this section, we include full qualitative results of using four variants in evaluation strategies ("Retrain"/"No-Retrain", MoRF/LeRF) for three different imputation operators (fixed value/Noisy Linear/GAN imputation). In Figure 13, the full



(a) Sample from CIFAR-10



(b) Sample from Food-101

Figure 12. Sample images from CIFAR-10 and Food-101 imputed with the three methods considered in this work for different percentages. The missing pixels are determined by the IG attribution method (in MoRF order). While the GAN leads to sharper images for the early percentage values, where the linearly imputed samples become more blurry. Artefacts are introduced for high missingness percentages (0.9) in GAN imputation, which may distort the results of the evaluation once again. Therefore, we decide to stick to the Noisy Linear Imputation that operates more stably.

plots of IG-family attribution methods using fixed value imputation are shown, while Figure 16 illustrates for the GB-based attribution methods. Figure 14 and Figure 17 show the evaluation results when using our Noisy Linear Imputation for IG- and GB-family attribution methods, respectively. From results, we see that using our Noisy Linear Imputation, the consistency between the evaluation rankings conducted in MoRF and LeRF with and without retraining increases, for instance in Figure 14 compared to Figure 13.

D. Additional Experiments on Food-101

D.1. Implementation Details

We trained a vanilla ResNet-50 (He et al., 2016) on Food-101 (Bossard et al., 2014). Concretely, we trained the model using the SGD optimizer. Additionally the model was trained with the initial learning rate of 0.01. The learning rate was reduced by factor of 0.1 after every 10 epochs. In total, we trained 40 epochs with a batch size of 32 and the model achieved the accuracy of 81.67% on the test set. To run the GAN imputation operator, we first trained a GAIN model on Food-101 as introduced in Appendix B. We used the hyper-parameters $\alpha = 100$ and $hr = 0.1$ and trained the GAIN model with the batch size of 32 for 100 epochs. We computed the eight explanations and run ROAD and ROAR evaluation using the same settings as introduced in Appendix C.1 for CIFAR-10.

D.2. Correlation Analysis

In Table 10, we show a full view of Spearman Correlation of rankings given by eight different evaluation strategies ("Retrain"/"No-Retrain", MoRF/LeRF, and fixed/Noisy Linear/GAN imputation) on Food-101. In the table, results marked in bold indicate the consistency of using three imputation operators. We observe that the consistency between the respective Retrain and No-Retrain methods is still very high, which confirms that the efficiency gains reported in the main paper can be realized for larger data sets. Consistency between MoRf/LeRF is improved (over fixed imputation) when using retraining, but decreases slightly when the No-Retraining approach is used. Because the curves are often very close on this dataset

		10	20	30	40	50	70	90
Retrain MoRF	fixed	74.94±0.57	75.42±0.45	75.62±0.24	75.16±0.50	74.95±0.45	73.73±0.48	65.18±0.85
	lin	69.72±0.49	68.10±0.34	67.28±0.34	67.32±0.22	67.52±0.22	66.46±0.54	60.37±0.51
	gan	74.78±0.31	73.16±0.22	72.02±0.03	71.40±0.23	70.72±0.30	68.44±0.43	59.37±0.44
No-Retrain MoRF	fixed	44.06±0.04	29.81±0.03	21.99±0.03	17.35±0.02	14.67±0.01	11.50±0.04	10.90±0.03
	lin	67.66±0.02	59.94±0.03	54.05±0.05	49.46±0.04	45.63±0.06	36.87±0.05	24.55±0.04
	gan	74.53±0.04	71.41±0.04	69.10±0.06	67.55±0.09	66.55±0.07	60.73±0.12	25.46±0.10
Retrain LeRF	fixed	80.88±0.14	81.34±0.15	81.41±0.01	81.36±0.14	81.34±0.11	80.95±0.01	76.86±0.34
	lin	81.41±0.10	81.67±0.18	81.88±0.16	81.56±0.13	81.31±0.22	79.89±0.23	72.83±0.36
	gan	81.05±0.22	80.99±0.15	80.14±0.16	79.25±0.18	78.24±0.22	74.92±0.15	68.69±0.21
No-Retrain LeRF	fixed	74.34±0.02	69.04±0.03	64.06±0.04	59.86±0.03	57.59±0.03	53.81±0.06	46.74±0.02
	lin	82.20±0.04	82.04±0.03	81.76±0.08	81.34±0.06	80.97±0.03	77.89±0.07	56.74±0.13
	gan	80.80±0.02	80.38±0.03	79.90±0.02	78.85±0.07	77.47±0.08	71.14±0.10	32.96±0.17

Table 6. Mean accuracy at each η by using IG-SG in all methods with standard deviations of five individual runs. For LeRF, the accuracy is at (1-η).

		10	20	30	40	50	70	90
Retrain MoRF	fixed	76.30±0.43	75.60±0.27	74.89±0.29	74.27±0.29	73.37±0.28	72.15±0.09	67.99±0.24
	lin	72.83±0.37	71.87±0.41	71.58±0.19	70.98±0.15	70.47±0.20	67.81±0.45	59.38±0.46
	gan	76.64±0.13	75.44±0.13	74.73±0.28	73.69±0.30	72.85±0.34	68.97±0.08	56.81±0.30
No-Retrain MoRF	fix	73.03±0.03	66.72±0.03	58.72±0.07	52.51±0.04	48.52±0.08	48.79±0.06	44.43±0.06
	lin	74.57±0.08	71.18±0.06	68.70±0.08	67.24±0.08	64.82±0.11	57.68±0.06	32.59±0.09
	gan	76.57±0.03	74.70±0.04	72.51±0.09	71.19±0.07	69.64±0.08	60.89±0.15	21.11±0.16
Retrain LeRF	fixed	72.39±0.39	71.76±0.41	71.21±0.30	70.26±0.50	69.83±0.22	68.32±0.45	63.29±0.56
	lin	72.86±0.24	71.63±0.27	70.67±0.42	70.08±0.30	69.82±0.22	68.10±0.18	60.12±0.34
	gan	75.97±0.27	74.73±0.27	73.41±0.24	72.74±0.34	72.20±0.28	69.89±0.26	57.57±0.24
No-Retrain LeRF	fixed	69.61±0.04	64.90±0.02	57.88±0.05	51.67±0.09	46.93±0.06	42.40±0.09	37.10±0.03
	lin	71.84±0.06	66.71±0.08	63.79±0.05	61.46±0.09	59.69±0.09	55.09±0.06	35.72±0.13
	gan	75.13±0.02	72.13±0.05	70.25±0.05	68.56±0.08	67.35±0.08	62.32±0.13	24.61±0.19

Table 7. Mean accuracy at each η by using GB-SG in all methods with standard deviations of five individual runs. For LeRF, the accuracy is at (1-η).

(in particular for the No-Retraining setup), small differences might already lead to a change in the ranking and the results are in general noisier than on CIFAR-10. In summary, we observe similar trends, although the consistency gain between MoRF/LeRF in No-Retrain is not as pronounced. Nevertheless, a perfect agreement between MoRF/LeRF might not be desirable.

D.3. Extended Figures

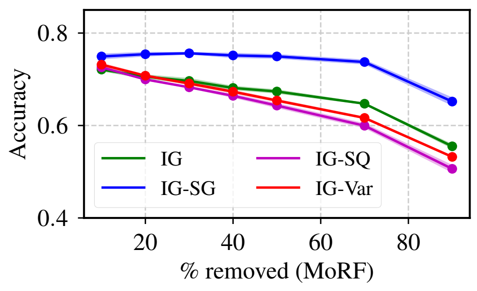
Full qualitative results of using four variants in evaluation strategies ("Retrain"/"No-Retrain", MoRF/LeRF) for three different imputation operators (fixed value/Noisy Linear/GAN imputation) are listed from Figure 19 to Figure 24. Figure 20 and Figure 23 show the evaluation results when using our Noisy Linear Imputation for IG- and GB-family attribution methods, respectively. From results, we see that using our Noisy Linear Imputation, the consistency between the evaluation results using "Retrain" and "No-Retrain" are more consistent compared to using the fixed value imputation. Therefore, retraining can be safely skipped by using our Noisy Linear Imputation.

Strategy	Retrain			No-Retrain		
	fixed†	lin	gan	fixed	lin*	gan
Time	3903±117 s	4686±2 s	6421±74 s	18.0±0.1 s	33.3±0.1 s	35.0±0.1 s
Relative	100 %	120 %	164 %	0.5 %	0.9 %	0.9 %

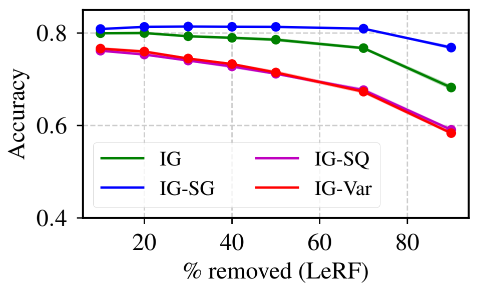
Table 8. Mean runtime (5 runs) for evaluating a single explanation method (IG) on three imputation operators. † refers to ROAR, and $\star$ to our ROAD.

		Retrain MoRF			No-Retrain MoRF			Retrain LeRF			No-Retrain LeRF		
		fixed†	lin	gan	fixed	lin*	gan	fixed	lin	gan	fixed	lin	gan
Retrain MoRF	fixed†	1.00 ±0.00											
	lin	0.68 ±0.02	1.00 ±0.00										
	gan	0.76 ±0.01	0.82 ±0.01	1.00 ±0.00									
No-Retrain MoRF	fixed	**0.15** ±0.01	0.38 ±0.02	0.23 ±0.01	1.00 ±0.00								
	lin*	0.66 ±0.01	**0.84** ±0.01	0.86 ±0.01	0.43 ±0.01	1.00 ±0.00							
	gan	0.65 ±0.01	0.62 ±0.01	0.84 ±0.01	0.14 ±0.01	0.78 ±0.01	1.00 ±0.00						
Retrain LeRF	fixed	**-0.01** ±0.01	0.48 ±0.02	0.28 ±0.02	0.66 ±0.00	0.47 ±0.02	0.13 ±0.01	1.00 ±0.00					
	lin	0.16 ±0.01	**0.61** ±0.01	0.34 ±0.01	0.78 ±0.01	0.50 ±0.01	0.10 ±0.01	0.87 ±0.01	1.00 ±0.01				
	gan	0.15 ±0.01	0.59 ±0.01	0.32 ±0.01	0.74 ±0.00	0.50 ±0.01	0.10 ±0.01	0.90 ±0.01	0.96 ±0.01	1.00 ±0.00			
No-Retrain LeRF	fixed	0.49 ±0.01	0.44 ±0.01	0.69 ±0.01	**0.01** ±0.00	0.60 ±0.00	0.77 ±0.00	**0.09** ±0.01	0.03 ±0.01	-0.03 ±0.00	1.00 ±0.00		
	lin	0.21 ±0.01	0.60 ±0.01	0.38 ±0.01	0.81 ±0.00	**0.58** ±0.01	0.22 ±0.01	0.85 ±0.00	**0.94** ±0.01	0.91 ±0.00	0.10 ±0.00	1.00 ±0.00	
	gan	0.05 ±0.01	0.47 ±0.01	0.17 ±0.01	0.69 ±0.00	0.36 ±0.00	-0.07 ±0.01	0.85 ±0.00	0.86 ±0.01	0.90 ±0.01	-0.14 ±0.00	0.79 ±0.00	1.00 ±0.00

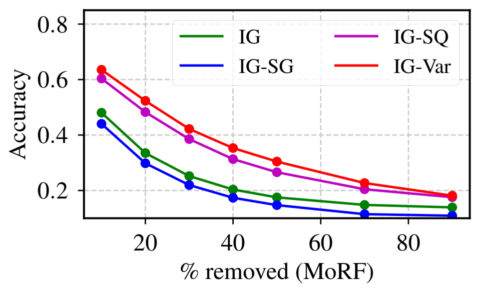
Table 9. **CIFAR-10**: Rank Correlations between all evaluation strategies used with standard deviations computed by considering the rankings obtained through five consecutive runs as independent. Results indicated in bold correspond to those reported in the main paper. The ROAR benchmark is marked by † and our ROAD by *.



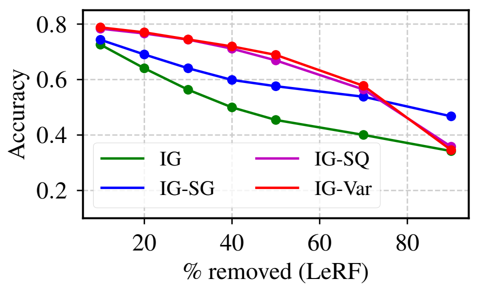
(a) MoRF, Retrain



(b) LeRF, Retrain

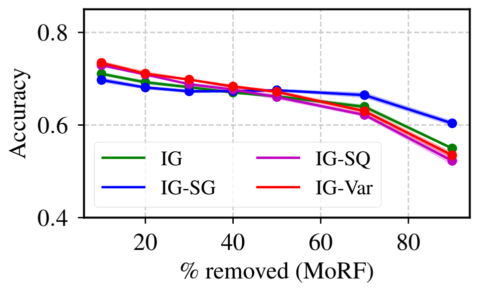


(c) MoRF, No-Retrain

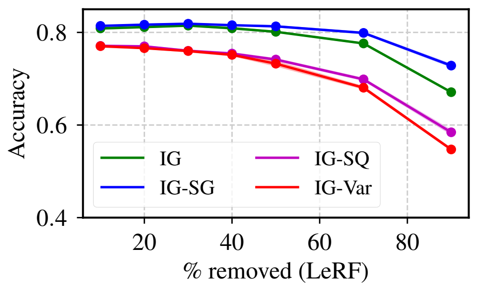


(d) LeRF, No-Retrain

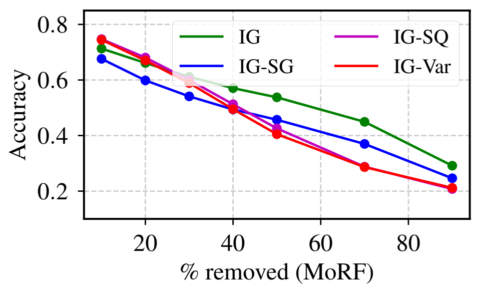
Figure 13. Consistency comparison using **Fixed Value** imputation on **IG**-based methods on CIFAR-10



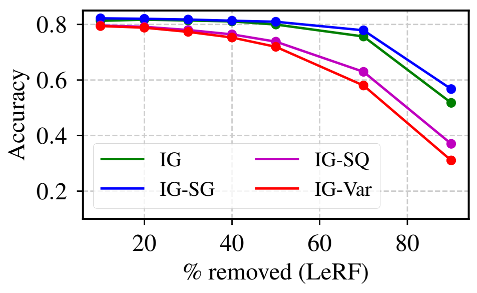
(a) MoRF, Retrain



(b) LeRF, Retrain

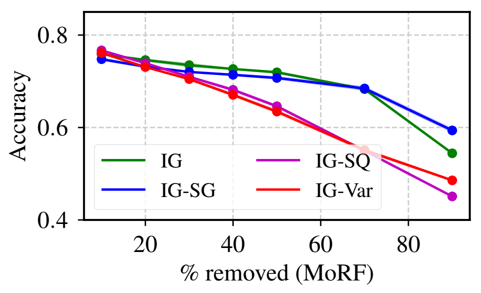


(c) MoRF, No-Retrain

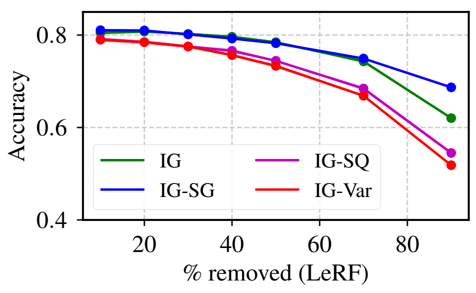


(d) LeRF, No-Retrain

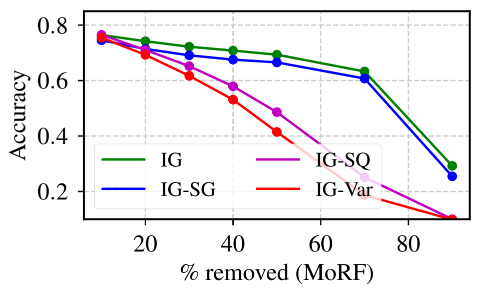
Figure 14. Consistency comparison using **Noisy Linear** imputation on **IG**-based methods on CIFAR-10



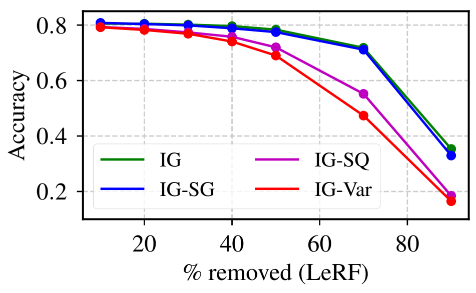
(a) MoRF, Retrain



(b) LeRF, Retrain

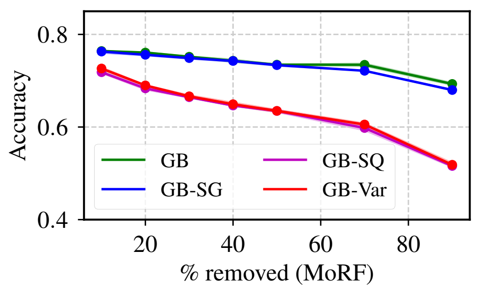


(c) MoRF, No-Retrain

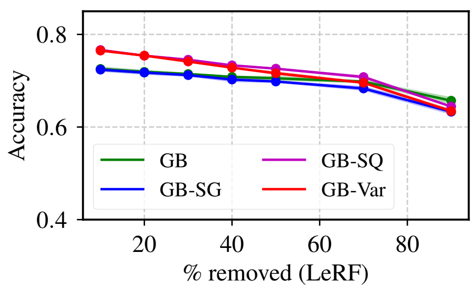


(d) LeRF, No-Retrain

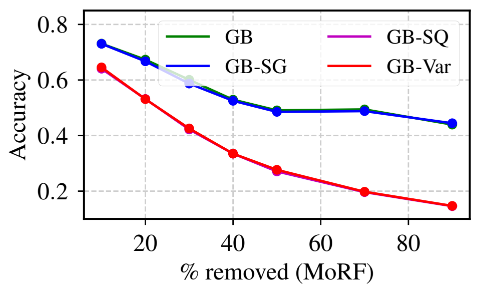
Figure 15. Consistency comparison using **GAN** imputation on **IG**-based methods on CIFAR-10



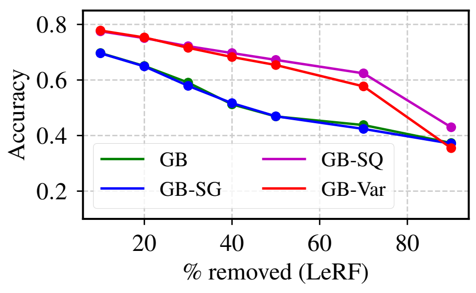
(a) MoRF, Retrain



(b) LeRF, Retrain

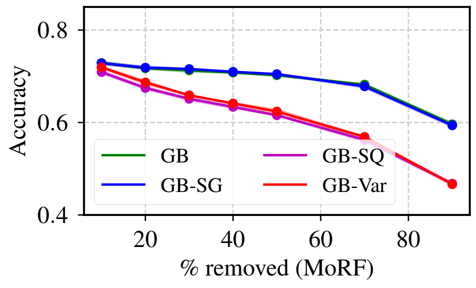


(c) MoRF, No-Retrain

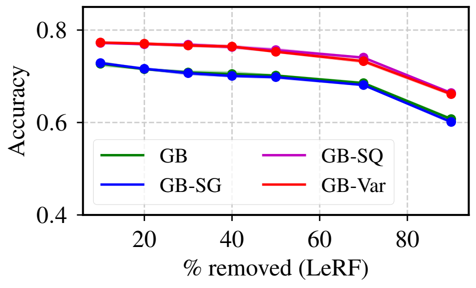


(d) LeRF, No-Retrain

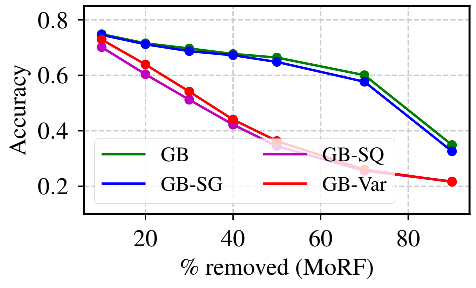
Figure 16. Consistency comparison using **Fixed Value** imputation on **GB**-based methods on CIFAR-10



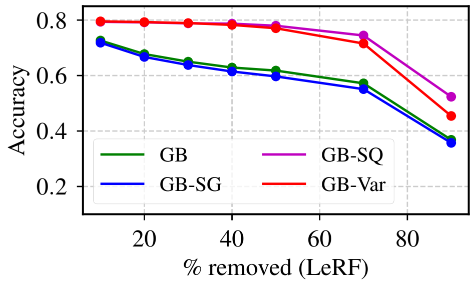
(a) MoRF, Retrain



(b) LeRF, Retrain

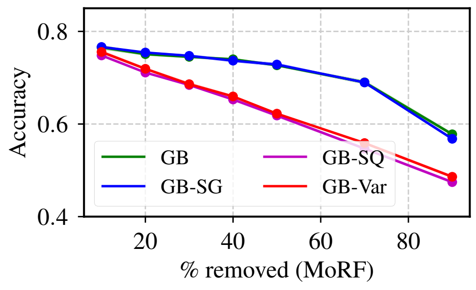


(c) MoRF, No-Retrain

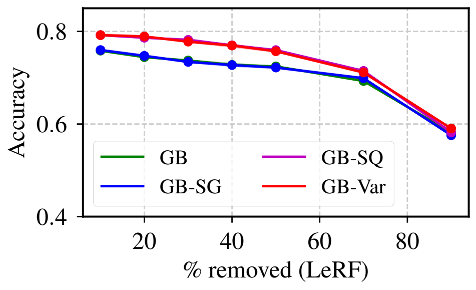


(d) LeRF, No-Retrain

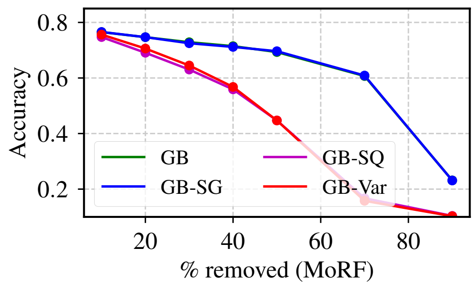
Figure 17. Consistency comparison using **Noisy Linear** imputation on **GB**-based methods on CIFAR-10



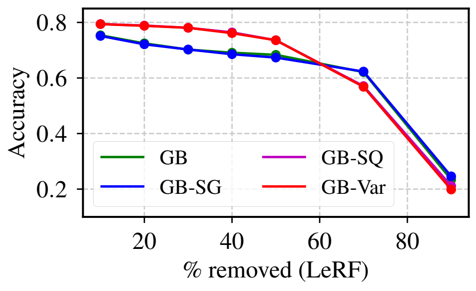
(a) MoRF, Retrain



(b) LeRF, Retrain



(c) MoRF, No-Retrain

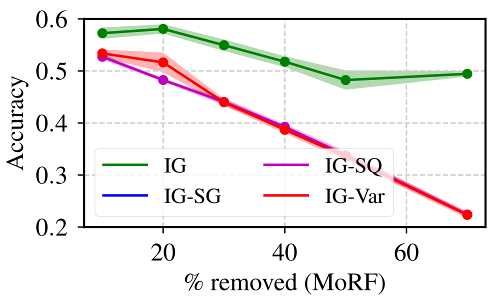


(d) LeRF, No-Retrain

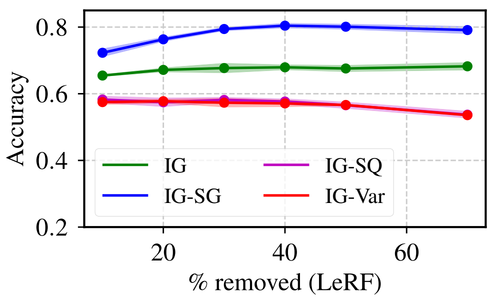
Figure 18. Consistency comparison using **GAN** imputation on **GB**-based methods on CIFAR-10

		Retrain MoRF			No-Retrain MoRF			Retrain LeRF			No-Retrain LeRF		
		fixed†	lin	gan	fixed	lin*	gan	fixed	lin	gan	fixed	lin	gan
Retrain MoRF	fixed†	1.00 ±0.00											
	lin	0.48 ±0.03	1.00 ±0.00										
	gan	0.50 ±0.04	0.79 ±0.03	1.00 ±0.00									
No-Retrain MoRF	fixed	**0.12** ±0.01	0.57 ±0.02	0.50 ±0.01	1.00 ±0.00								
	lin*	0.61 ±0.01	**0.81** ±0.02	0.67 ±0.04	0.31 ±0.01	1.00 ±0.00							
	gan	0.74 ±0.01	0.79 ±0.02	**0.67** ±0.04	0.35 ±0.01	0.86 ±0.00	1.00 ±0.00						
Retrain LeRF	fixed	**-0.26** ±0.02	0.41 ±0.02	0.30 ±0.02	0.53 ±0.01	0.10 ±0.01	0.11 ±0.01	1.00 ±0.00					
	lin	-0.40 ±0.02	**0.26** ±0.04	0.19 ±0.04	0.30 ±0.03	-0.05 ±0.01	0.09 ±0.01	0.83 ±0.01	1.00 ±0.00				
	gan	-0.18 ±0.01	0.46 ±0.04	**0.32** ±0.04	0.50 ±0.03	0.13 ±0.02	0.14 ±0.03	0.89 ±0.02	0.83 ±0.01	1.00 ±0.00			
No-Retrain LeRF	fixed	0.79 ±0.02	0.79 ±0.03	0.63 ±0.05	**0.32** ±0.01	0.85 ±0.00	0.89 ±0.00	**0.02** ±0.01	-0.15 ±0.02	0.10 ±0.03	1.00 ±0.00		
	lin	-0.28 ±0.02	0.35 ±0.02	0.28 ±0.04	0.46 ±0.00	**-0.03** ±0.00	-0.06 ±0.00	0.89 ±0.01	**0.81** ±0.02	0.87 ±0.01	-0.11 ±0.00	1.00 ±0.00	
	gan	-0.45 ±0.02	-0.08 ±0.03	-0.04 ±0.04	0.23 ±0.00	-0.37 ±0.00	**-0.44** ±0.00	0.58 ±0.01	0.61 ±0.01	**0.54** ±0.00	-0.41 ±0.00	0.70 ±0.00	1.00 ±0.00

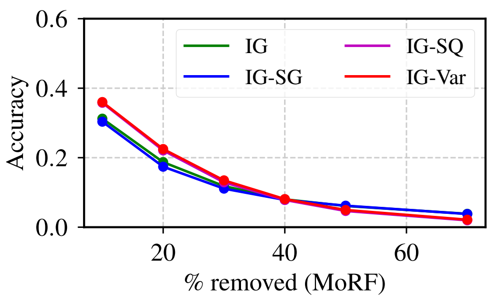
Table 10. **Food-10**: Rank Correlations between all evaluation strategies used with standard deviations computed by considering the rankings obtained through five consecutive runs as independent. The ROAR benchmark is marked by † and our ROAD by *. Bold results highlight the consistency between Retrain and No-Retrain (still very high) as well as MoRF and LeRF evaluation strategies using different imputation operators (fair increase when using Noisy Linear and GAN imputations instead of fixed imputation in "Retrain", decrease in "No-Retrain").



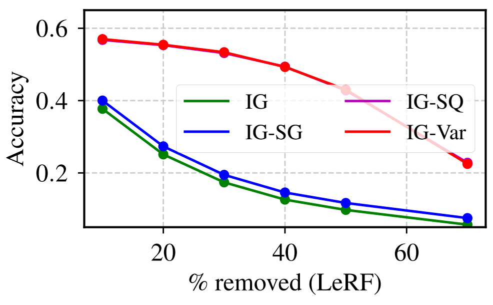
(a) MoRF, Retrain



(b) LeRF, Retrain

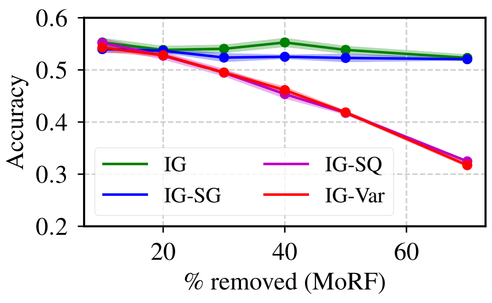


(c) MoRF, No-Retrain

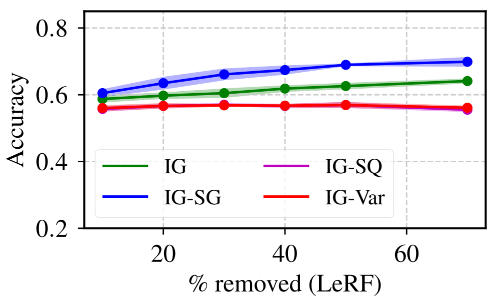


(d) LeRF, No-Retrain

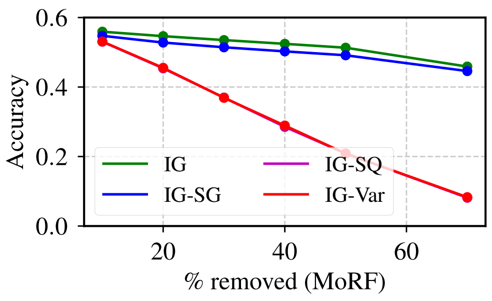
Figure 19. Consistency comparison using **Fixed Value** imputation on **IG**-based methods on Food-101.



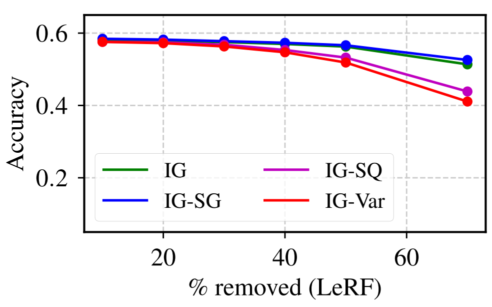
(a) MoRF, Retrain



(b) LeRF, Retrain

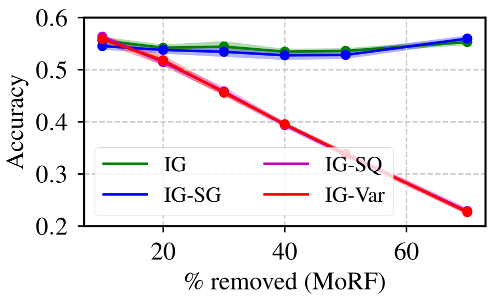


(c) MoRF, No-Retrain

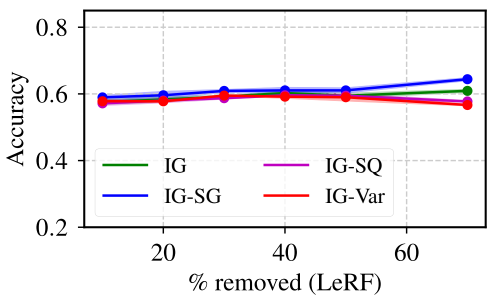


(d) LeRF, No-Retrain

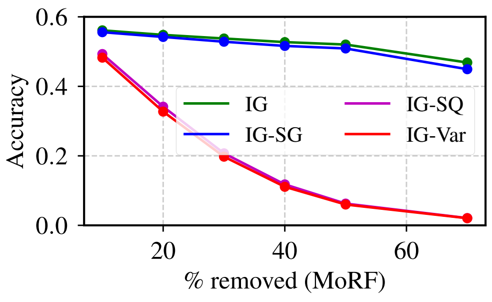
Figure 20. Consistency comparison using **Noisy Linear** imputation on **IG**-based methods on Food-101.



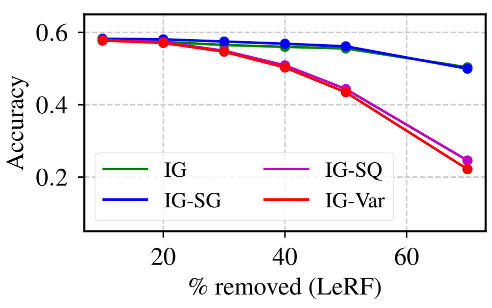
(a) MoRF, Retrain



(b) LeRF, Retrain

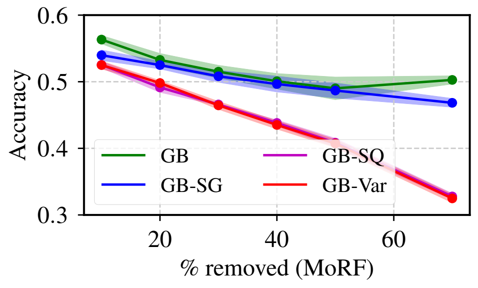


(c) MoRF, No-Retrain

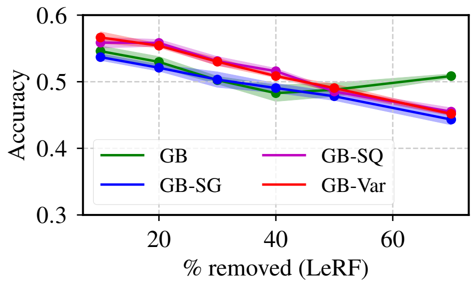


(d) LeRF, No-Retrain

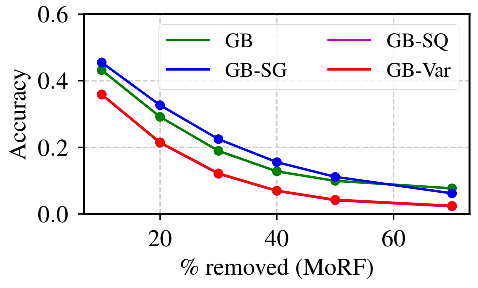
Figure 21. Consistency comparison using **GAN** imputation on **IG**-based methods on Food-101.



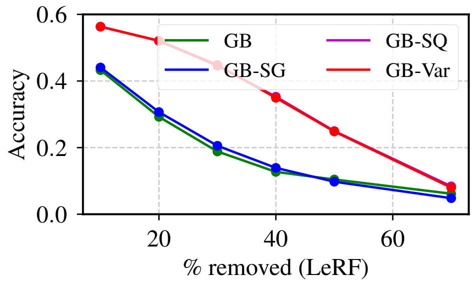
(a) MoRF, Retrain



(b) LeRF, Retrain

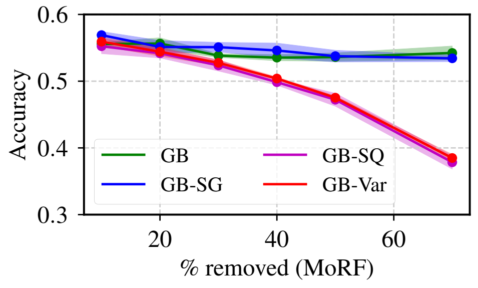


(c) MoRF, No-Retrain

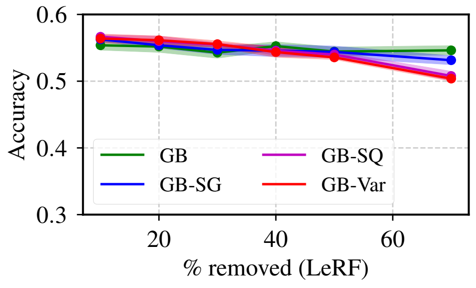


(d) LeRF, No-Retrain

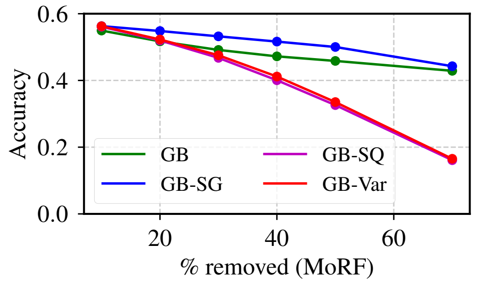
Figure 22. Consistency comparison using **Fixed Value** imputation on **GB**-based methods on Food-101.



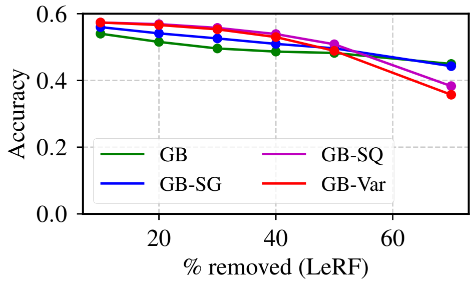
(a) MoRF, Retrain



(b) LeRF, Retrain

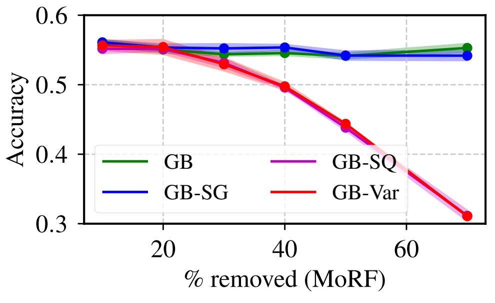


(c) MoRF, No-Retrain

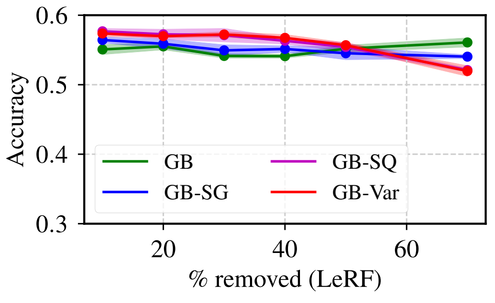


(d) LeRF, No-Retrain

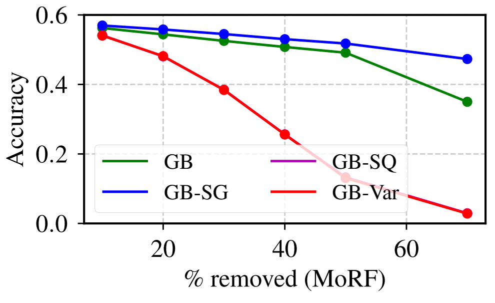
Figure 23. Consistency comparison using **Noisy Linear** imputation on **GB**-based methods on Food-101.



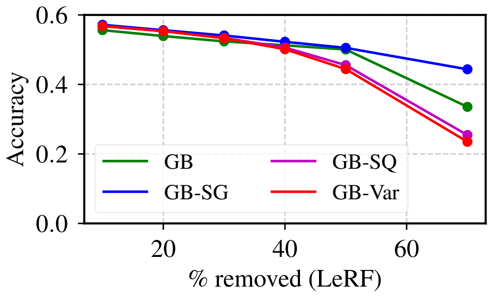
(a) MoRF, Retrain



(b) LeRF, Retrain



(c) MoRF, No-Retrain



(d) LeRF, No-Retrain

Figure 24. Consistency comparison using **GAN** imputation on **GB**-based methods on Food-101.

A.2.4 Relational Local Explanations

Publication: At the moment of writing, the paper is in submission to an international machine learning conference.

Contribution: I developed all major parts of the explainability framework and the implementations of the relational local explanation framework. I also performed the experiments and wrote most parts of the paper. Gjergji Kasneci contributed to the paper by challenging and improving early ideas, formalizations, and the analysis of results. All co-authors helped revise the final manuscript.

Relational Local Explanations

Vadim Borisov[*]
University of Tübingen

Gjergji Kasneci
University of Tübingen

Abstract

The majority of existing post-hoc explanation approaches for machine learning models produce independent per-variable feature attribution scores, ignoring a critical characteristic, such as the inter-variable relationship between features that naturally occurs in visual and textual data. In response, we develop a novel model-agnostic and permutation-based feature attribution algorithm based on the relational analysis between input variables. As a result, we are able to gain a broader insight into machine learning model decisions and data. This type of local explanation measures the effects of interrelationships between local features, which provides another critical aspect of explanations. Experimental evaluations of our framework using setups involving both image and text data modalities demonstrate its effectiveness and validity.

1 Introduction

The increasing reliance on machine learning (ML) models in various domains of our daily life has brought a need for explaining the inner workings and decision-making processes of these models [1]. This is particularly relevant for deep convolutional neural networks (CNNs) in visual domains, which have demonstrated superior performance on tasks such as object detection [2], segmentation [3], and classification [4]. Also, in the natural language processing (NLP) domain, self-attention models [5], specifically deep Transformer-based models, have achieved state-of-the-art results on tasks such as text summarization [6] and sentiment analysis [7].

As a result, it is necessary to have confidence that black-box ML models are functioning as intended, and explanations that include *inter-variable relational information* can help achieve this. Moreover, the interpretability of ML models is a vital aspect for numerous applications, particularly those involving life-critical uses such as healthcare and autonomous driving [8, 9].

Furthermore, in accordance with the General Data Protection Regulation (GDPR) [10] and California Consumer Privacy Act (CCPA) [11], it is essential for real-world applications to not only provide accurate and reliable predictions but also to provide transparent and easily understood explanations for automated decision systems. Additionally, there is a practical need for model-agnostic feature methods that can be used with any machine learning system. Last but not least, from a practical industrial perspective, there is a need for model-agnostic feature methods which can work with any ML system [12].

Motivation. Although numerous feature attribution approaches exist, the vast majority of them work with each input variable *independently*, ignoring the crucial property such as the relationship that intrinsically exists in the many homogeneous data formats such as visual and textual.

Another issue with the state-of-the-art feature attribution approaches is that many local explanation methods "corrupt" a data sample to obtain local approximations of it [13, 14], as a result, it leads to the out-of-the distribution problem [15]. Further discussion of this topic is provided in Section 2 and Section 5.

[*]Corresponding author: vadim.borisov@uni-tuebingen.de

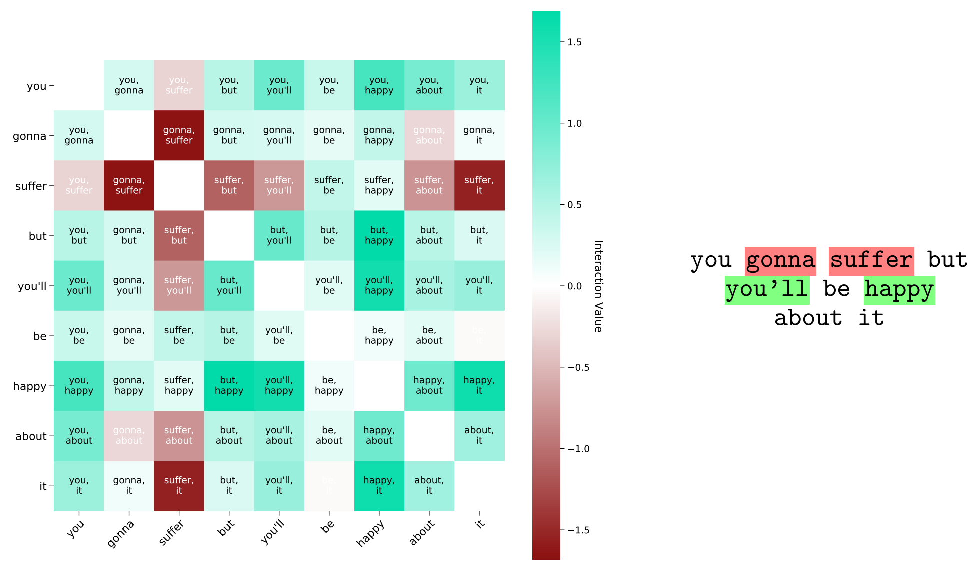


Figure 1: An example of the relational local explanation (*left*) and standard local explanation (*right*) for textual data from the proposed RLE framework, where green color indicates positive influence and red negative. It can be seen that the relational local explanation allows the analysis of the pairwise influence of each word. For the task we select a pre-trained `DistilBERT` model [16] for the sentiment analysis task. For more results, please refer to the Sec. 4 and Appendix A.

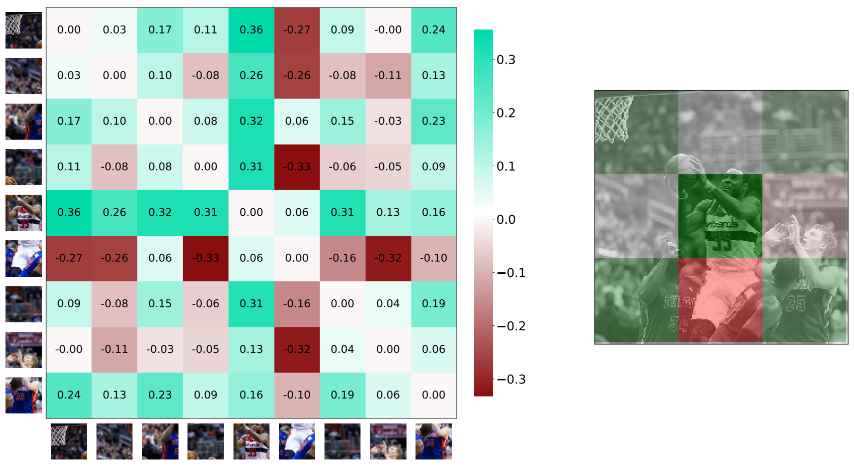
In response, we propose a new framework for post hoc feature attribution that provides *relational local explanations*. Our framework represents homogeneous data as a graph and leverages the edge information to identify the relationship between features.

The proposed approach presents a double-view on the feature attributions: (1) General local explanations as coefficients of how particular variables (words or a group of pixels) influence the decision of the given ML model positively or negatively, w.r.t other variables. (2) Relational local explanations in the form of attention matrices, where the relationship between each input variable and other variables is represented as a coefficient. This type of explanation answers an important question - *How strongly is this variable related to all other variables?* By that, we add another layer of depth to the explanation.

Contributions. Below, we list the main contributions of our work are:

- We highlight the importance of relational interactions between input features for local explanations. Since visual and textual data types are "compositional" per nature i.e. the "regional" information between variables naturally exists, it is crucial not only to understand what variable is important but also to spotlight and quantify the most critical combinations of them in a given data sample.
- We develop a novel model-agnostic local feature attribution technique - coined relational local explanations (RLE) - and formally describe it. To the best of our knowledge, this is the first model-agnostic local explanation algorithm based on the relationships between input variables.
- We extensively evaluate the proposed approach on image and text datasets and empirically show that it produces explanations that are superior to those produced by state-of-the-art attribution techniques.
- We open-sourced the RLE implementation `https://github.com/unnir/rle`.

The remainder of this work is organized as follows. In Section 2, we give a short overview of the related methods for explaining machine learning models. Section 3 presents the proposed RLE algorithm. After, in Section 4 we visually and empirically compare our algorithm against other state-of-the-art feature attribution approaches. Section 5 discusses the properties and limitations of the proposed method before we conclude in Section 6.



0.00	0.03	0.17	0.11	0.36	-0.27	0.09	-0.00	0.24
0.03	0.00	0.10	-0.08	0.26	-0.26	-0.08	-0.11	0.13
0.17	0.10	0.00	0.08	0.32	0.06	0.15	-0.03	0.23
0.11	-0.08	0.08	0.00	0.31	-0.33	-0.06	-0.05	0.09
0.36	0.26	0.32	0.31	0.00	0.06	0.31	0.13	0.16
-0.27	-0.26	0.06	-0.33	0.06	0.00	-0.16	-0.32	-0.10
0.09	-0.08	0.15	-0.06	0.31	-0.16	0.00	0.04	0.19
-0.00	-0.11	-0.03	-0.05	0.13	-0.32	0.04	0.00	0.06
0.24	0.13	0.23	0.09	0.16	-0.10	0.19	0.06	0.00

Figure 2: An example of the relational local explanation (*left*) and standard local explanation (*right*) for visual data from the proposed RLE framework where green color indicates positive influence, and red negative. The relational local explanation can be used for a deeper feature analysis of the image data. For the task we select a pre-trained ResNet-50 model [4] and an image with a class `basketball` from the ImageNet data set [24], e.g., to uncover a combination of patches that is the most important to a model. For more results, please refer to Appendix. 4.

2 Related Work

In recent years, there have been several studies that have focused on methods for explaining feature interactions and adjacency. Lundberg et al. [17] propose an efficient local explanation method based on the SHAP framework [18] for decision tree-based models, which allows for the direct measurement of local feature interaction effects. Cui et al. [19] propose a probabilistic estimation method to assess the joint effect of two input features and the sum of their marginal effects in order to evaluate global feature pairwise interactions.

A number of studies have also explored feature interaction approaches specifically for deep neural networks (DNNs). For example, Greenside et al. [20] explore interactions between variables using deep feature interaction maps by calculating the difference between the attributions of two variables. Singh et al. [21] present the generalization of the Contextual Decomposition [22] to explain interactions for dense DNNs and CNNs.

More recently, Janizel et al. [23] propose an efficient method for feature interaction local explanation for DNNs called Integrated Hessians (IH). This method is based on an enhancement of the Integrated Gradients (IG) approach [14] and has been shown to produce trustworthy results. However, from a practical perspective, the Hessian matrix is significantly larger than gradient matrices and requires a sufficient memory size.

Limitations of prior approaches. Despite the progress made in understanding feature relationships through previous approaches, a major limitation of these methods is that they are often tied to specific model architectures or data types, making them not fully model agnostic. In addition, perturbation-based algorithms such as LIME [13], SHAP [18], and IG [14] rely on altering the data sample in order to provide explanations, while our method aims to preserve as much information as possible by only altering the global structure. We discuss this issue in Section 5.

3 Relational Local Explanation (RLE) Framework

This section introduces the Relational Local Explanation (RLE) algorithm by first discussing its main components. In addition, we present how the RLE approach can be utilized for visual and textual data modalities.

Algorithm 1 Relational Local Explanations (RLE)

```
Require: ML model $f$, Instance to explain $\boldsymbol{x}_o$, Number of permutations $n$
  $\mathcal{X}' \leftarrow \{\}$                                  ▷ New auxiliary data set
  for $i \in \{1, 2, 3, ..., n\}$ do
      $\boldsymbol{x}_i^p \leftarrow permute(\boldsymbol{x}_o)$          ▷ Replace and shuffle the instance to explain
      $\mathcal{G}_i \leftarrow Graph(\boldsymbol{x}_i^p)$                ▷ Get the graph structure of patches
      $\mathbf{A}_i \leftarrow AdjacencyMatrix(\mathcal{G}_i)$            ▷ Get the adjacency matrix
      $\boldsymbol{x}' \leftarrow Lower(\mathbf{A}_i)$                    ▷ Get the lower triangle
      $\mathcal{X}' \leftarrow \mathcal{X}' \cup \langle x', f(\tilde{x}_i) \rangle$
  end for
  $w \leftarrow LinearModel(\mathcal{X}')$                         ▷ Train an explainable-by-design surrogate model
  return $w$
```

3.1 Formal definitions

Before proceeding to the description of the proposed method, we introduce central definitions of our study. The definitions are based on existing works [18, 14].

Definition 1 (Local Explanation) *A feature attribution function can be seen as $\phi(f, \boldsymbol{x}, c_{\boldsymbol{x}}) \in \mathbb{R}^n$, where $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is a black-box model and $\boldsymbol{x} \in \mathbb{R}^n$ is an input sample belonging to a class $c_{\boldsymbol{x}} \subset \mathbb{R}$. The output of ϕ is an explanation representation vector $\mathbf{e}_{\boldsymbol{x}} \in \mathbb{R}^n$.*

Each element of $\mathbf{e}_{\boldsymbol{x}}$ is an importance score corresponding to a feature value in $\boldsymbol{x}$. A large positive or negative value in $\mathbf{e}_{\boldsymbol{x}}$ indicates that a corresponding feature greatly influences the outcome. Features with values close to zero in $\mathbf{e}_{\boldsymbol{x}}$ have little impact. Note that there are explanation methods that do not require a class specification; thus, for simpler and more general notation, a feature scoring function has the form $\phi(f, \boldsymbol{x})$.

Taking the description of local explanations, we can extend it to the definition of relational local explanations.

Definition 2 (Relational Local Explanation) *A relational local explanation function can be seen as $\Psi(f, \boldsymbol{x}, c_x) \in \mathbb{R}^{n \times n}$, where $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is a black-box model as above and $\boldsymbol{x} \in \mathbb{R}^n$ is an input sample belonging to a class $c_{\boldsymbol{x}} \subset \mathbb{R}$. The output of the Ψ is an relational explanation representation in a form of a adjacency matrix $\mathbf{A}_{\boldsymbol{x}}$.*

The relational local explanation matrix $\mathbf{A}_{\boldsymbol{x}}$ contains in each cell $\mathbf{A}_{\boldsymbol{x}}(i, j)$ the relational interaction between the i'th and j'th input feature. Note that $\mathbf{A}_{\boldsymbol{x}}$ is symmetric, i.e., $\mathbf{A}_x = \mathbf{A}_{\boldsymbol{x}}^\top$, and thus, the mean value of column or row elements corresponds to average local importance for the corresponding feature. Formally, let $\bar{\mathbf{A}}_{\boldsymbol{x}}$ denote the vector of mean values of the rows of matrix $\mathbf{A}_x$. Then:

$$\bar{\mathbf{A}}_x \mathrel{\hat{=}} \mathbf{e}_{\boldsymbol{x}}. \tag{1}$$

The symmetry property of relational local explanations is based on the assumption that the association between two variables has to be symmetrical. This was also indicated in the previous related works [23].

3.2 RLE : The proposed framework

Our approach follows common strategies for the generation local explanation proposed in LIME [13], SHAP [18], and Anchors [25], since they have a solid theoretical foundation [26] and established reputation in the ML community [1, 27]. The main idea of the RLE algorithm is to generate n local permutations of a data sample to explain, then construct corresponding graph representations and adjacency matrices of the relationships between input features from the shuffled data. Thus we obtain a new data set of local relations between features. Next, a linear model (that is explainable by design) is fitted to the new data set - using information from the adjacency matrices to get the corresponding coefficients, which can be utilized for the relational local explanation.

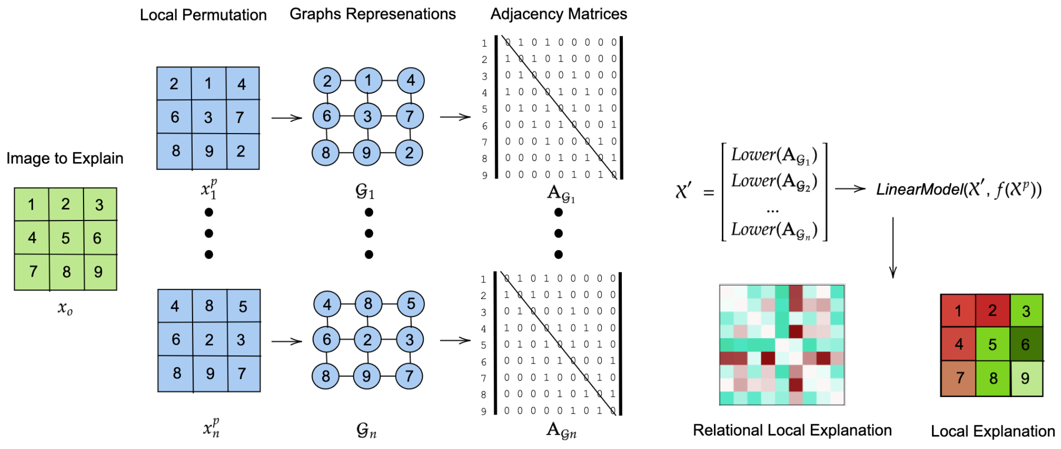

Local Permutation
Graphs Represenations
Adjacency Matrices
Image to Explain
x_o
x_1^p
x_n^p
$\mathcal{G}_1$
$\mathcal{G}_n$
$\mathbf{A}_{\mathcal{G}_1}$
$\mathbf{A}_{\mathcal{G}n}$
$\mathcal{X}' = [Lower(\mathbf{A}_{\mathcal{G}_1}), Lower(\mathbf{A}_{\mathcal{G}_2}), \ldots, Lower(\mathbf{A}_{\mathcal{G}_n})]$
LinearModel($\mathcal{X}'$, $f(\mathcal{X}^p)$)
Relational Local Explanation
Local Explanation

Figure 3: A relational local explanation of a data sample given a vision model using the RLE algorithm. Where f is a black-box machine learning model to explain, $\boldsymbol{x}_o$ is a sample of interest, $\mathcal{G}_i$ is a graph of representation of a perturb image sample x_i^p.

Formally, for a given black-box model $f : \mathcal{X} \to \mathcal{Y}$ with $\mathcal{X} \subseteq \mathbb{R}^n$, $\mathcal{Y} \subset \mathbb{R}$, we may learn an interpretable *surrogate* model g, which is a local approximation of f for a given perturbation of a particular input $\boldsymbol{x}_o \in \mathcal{X}$. For this purpose, we first divide a data sample (image, text) into discrete elements of pixel patches for visual data, and groups of words for textual data. Then the chosen data sample $\boldsymbol{x}_o$ is perturbed n times to generate $\boldsymbol{x}_{oi}^p, i = 1..n$ and we randomly replace a single element (i.e., patch/group) p_{oi} from $\boldsymbol{x}_o$ with another randomly selected element from $\boldsymbol{x}_0$. After an undirected graph representation of the shuffled sample is received $\mathcal{G}_{x_o}^p$, where each vertex is a discrete element p (e.g., superpixel or word), and the edge is the connection between them. Further, an adjacency matrix $\mathbf{A}_{\mathcal{G}_i}$ for each $\boldsymbol{x}_i^p$ is obtained. Since *adjacency matrices* for undirected graphs are symmetric, only the lower triangle is utilized $Lower(\mathbf{A}_{\mathcal{G}_i})$ for the next step. The key idea is to permute a data sample and keep local features since the strong perturbation may lead to the out-of-distribution problem [15]; we examine this issue in detail in Section 5.

These procedures yield a new data set $\mathcal{X}' = \{Lower(\mathbf{A}_{\mathcal{G}_i}), f(\boldsymbol{x}_i^p)\}_{i=1}^n$. We then learn a sparse linear regression $g_{\boldsymbol{w}_{\boldsymbol{x}_o}}(\boldsymbol{x}^p) = \boldsymbol{w}_{\boldsymbol{x}_0}^\top \boldsymbol{x}^p$ using the local data set $\mathcal{X}'$ by optimizing the following loss function with $\Omega(\cdot)$ as a measure of complexity.

$$w_{\boldsymbol{x}_0} = \underset{\boldsymbol{w}}{\operatorname{argmin}}\, \mathcal{L}(f, g) + \Omega(\boldsymbol{w}), \tag{2}$$

where $\mathcal{L}(f, g)$ is the mean squared loss,

$$\mathcal{L}(f, g) = \frac{1}{n} \sum_{i=1}^{n} \Big[f(x_i^p) - g(\boldsymbol{x}_i^p, \boldsymbol{x}_0) \Big]^2. \tag{3}$$

The RLE algorithm yields $g_{\boldsymbol{w}_{\boldsymbol{x}_0}}(\boldsymbol{x}')$, which approximates the complex model $f(\boldsymbol{x}')$ around $\boldsymbol{x}_0$. In case g is a linear model, the components of the weight vector $\boldsymbol{w}_{\boldsymbol{x}_0}$ indicate the relative influence of the relationship between features values of $\boldsymbol{x}_0$ based the sample $\mathcal{X}'$ and can be used as the relational local explanation of $f(\boldsymbol{x}_0)$. The full approach is summarized in Algorithm 1.

The following subsections discuss how the proposed algorithm can be adapted to the visual and textual data modalities.

3.3 Relational local explanations for multidimensional visual data

In the case of visual data, we divide an image into patches to disrupt the spatial layout of local image regions, this is a common approach for the local explanation methods [13, 18]. Formally, given an input image I, we first uniformly partition the image into $N \times N$ sub-regions denoted by $R_{i,j}$,

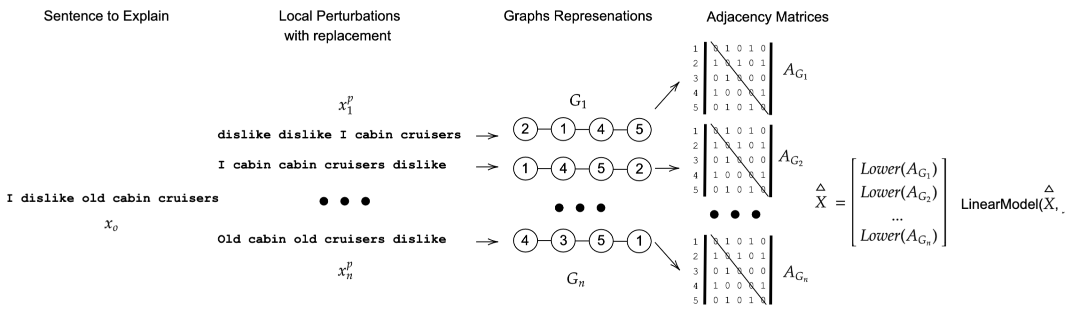

Sentence to Explain
Local Perturbations with replacement
Graphs Represenations
Adjacency Matrices
x_1^p
dislike dislike I cabin cruisers
I cabin cabin cruisers dislike
I dislike old cabin cruisers
x_o
Old cabin old cruisers dislike
x_n^p
G_1
G_n
A_{G_1}
A_{G_2}
A_{G_n}
$\overset{\triangle}{X} = [Lower(A_{G_1}), Lower(A_{G_2}), ..., Lower(A_{G_n})]$
LinearModel($\overset{\triangle}{X}$,

Figure 4: Relational local explanation of a data sample given a textual model using the RLE algorithm. Where f is a black-box machine learning model to explain, $\boldsymbol{x}_o$ is a sentence of interest, $\mathcal{G}_i$ is a graph of representation of a perturb sentence x_i^p.

where i and j are the horizontal and vertical indices respectively and $1 \leq i, j \leq N$. The procedure is presented in Fig. 3. Following that, a single patch $R_{i,j}$ represents a feature for the RLE algorithm.

Note that obtained graph representations for patched images do not consider the neighbor direction, e.g., a patch connected from the top, bottom, right, or left. To include the directional information, we may use a simple one-hot-encoding encoding technique for the adjacency matrix $\mathbf{A}_{\mathcal{G}_i}$. In practice, we observe that a meaningful relational local explanation can be constructed even without the one-hot-encoding step.

We argue that local details are much more important than a global structure for fine-grained image recognition, as it is these details that distinguish between different classes. In most cases, various fine-grained categories tend to share similar global structures and vary only in specific local details [28]; therefore, by random permutation, a given computer vision black-box model is forced to focus on the local details, and it favorites the most distinguished areas. Multiple studies support our argument: the jigsaw puzzle pretext task for Self-Supervised Learning (SSL) approaches [29], a regularization scheme for Variational Autoencoders (VAEs) [30], and last but not least for Vision Transformers (ViTs) an image is divided into patches as well [31, 32]. Besides, graph-based representation of visual data is a common approach for many downstream tasks [33].

3.4 Relational local explanations for textual data

For relational local explanations of the textual models, in particulate self-attention-based models [5], we represent a sentence as a graph $\mathcal{G}_i$, where each word is expressed as a node. Then, as we presented before, we permute the sentence by chaining the word order with replacement n times. The whole procedure is illustrated in Fig. 4. This operation allows learning bidirectional context for a word. As a result, each position grasps directional context from both “directions”.

The permutation idea was successfully used for training an XLNet [34] and GReaT [35] Transformer models; thus, it shows that transformer models are able to understand the semantics of a shuffled sentence. Furthermore, multiple studies [36, 37] made the same observation - a sentence can be seen as a graph, where words correspond to nodes and the computation of an attention score is the assignment of a weight to an edge between two worlds.

4 Experiments

We evaluate the RLE framework on several data sets by visually comparing them with state-of-the-art explanation methods with the goal of ensuring that our method fulfills its purpose. First, in Section 4.1 we present a visual comparison of state-of-the-art methods and the proposed framework. After, we compare local explanation for a textual data given a pre-trained DistilBERT model [16], furthermore, we demonstrate relational local explanation for a sentence in Section 4.2. Next, Section 4.3 presents a quantitative benchmark analysis of the RLE method with a comparison to selected baselines. For more visual experiments and detailed reproducibility details, please refer to the supplemental materials.

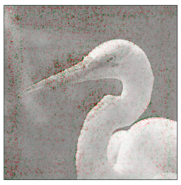
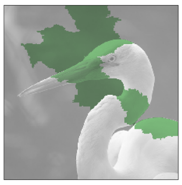
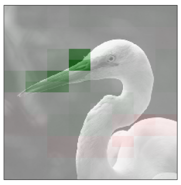
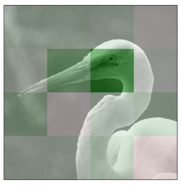
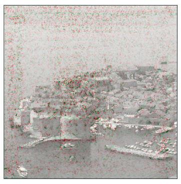
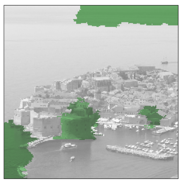
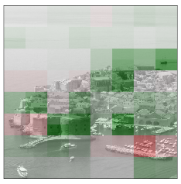
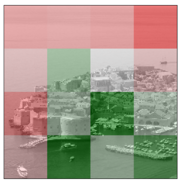
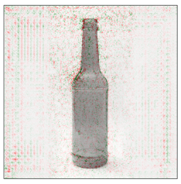
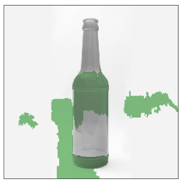
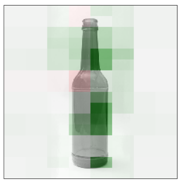
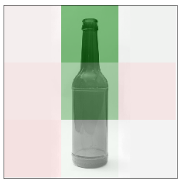
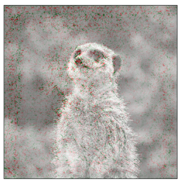
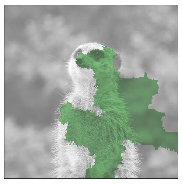
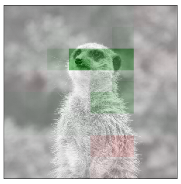
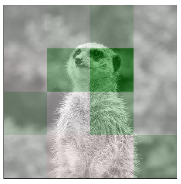
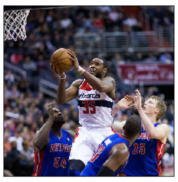
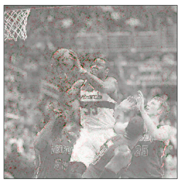
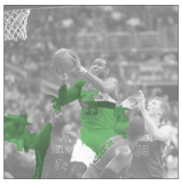
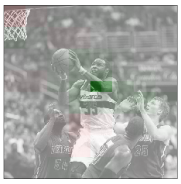
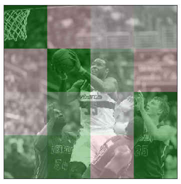
Original	IG [14]	LIME [13]	SHAP [18]	RLE (Ours)
				
				
				
				
				

Table 1: A Comparison of different state-of-the-art local explanation methods for a pre-trained ResNet-50 model [4] given random samples from the ImageNet data set [24]. Name of the original classes according to the selected model (from top to bottom): `American egret`, `seashore`, `bottle`, `marmot`, and `basketball`.

4.1 Visual analysis on image data

In our first experiment, a qualitative visual evaluation on images from the ImageNet data set [24] is performed for selected baseline: IG [14], LIME [13], SHAP [18], and the proposed RLE framework. We color the explanations from all baselines to illustrate that they distinguish between input variables that positively (green) and negatively (red) contribute to the CNN model estimations for a given class. The results are summarized in Table. 1. Also, an example of a relational local explanation for an image is depicted in Fig. 2.

4.2 Comparison on textual data

For the evaluation of the proposed method on the textual data, we utilize a pre-trained self-attention-based DistilBERT model for the sentiment analysis task [16] from the open-source Hugging Face library [38]. The results are summarized in the Table 2. Also, we demonstrate relational local explanations analysis using the RLE framework (Fig. 1) for the same transformer model and compare it to results from the IH method in Fig. 5.

Method	Local Explanation
IG [14]	You gonna suffer but you'll be happy about it
LIME [13]	You gonna suffer but you'll be happy about it
SHAP [18]	You gonna suffer but you'll be happy about it
IH [23]	You gonna suffer but you'll be happy about it
RLE (ours)	You gonna suffer but you'll be happy about it
IG [14]	You might be interested this product performs well
LIME [13]	You might be interested this product performs well
SHAP [18]	You might be interested this product performs well
IH [23]	You might be interested this product performs well
RLE (ours)	You might be interested this product performs well
IG [14]	The idea is nicely presented, but it has some limitations
LIME [13]	The idea is nicely presented, but it has some limitations
SHAP [18]	The idea is nicely presented, but it has some limitations
IH [23]	The idea is nicely presented, but it has some limitations
RLE (ours)	The idea is nicely presented, but it has some limitations

Table 2: A comparison of state-of-art feature attribution approaches to the presented RLE algorithm. given a pre-trained DistilBERT model [16] for the sentiment analysis task. We highlight the most important words according to each feature attribution method. For more results, please refer to the supplementary materials.

4.3 Quantitative comparison

In order to quantitatively evaluate our novel explanation framework, we utilize the well-accepted measure in the ML community - Iterative Removal Of Features (IROF) [39]. The IROF measure is fully described in the supplementary materials. The full definition of the mesure is in the Appendix C. This technique was featured in multiple studies before [40]. We compare against this study baselines: IG [14], LIME [13], and SHAP [18]. We also introduce the random baseline as the "sanity check", which assigns variable importance randomly. Notably, the authors of [41, 42] show that this primitive baseline can outperform some of the commonly used explanation approaches based on saliency maps in ablation tests. Results are in Table 3.

4.4 Reproducibility Details

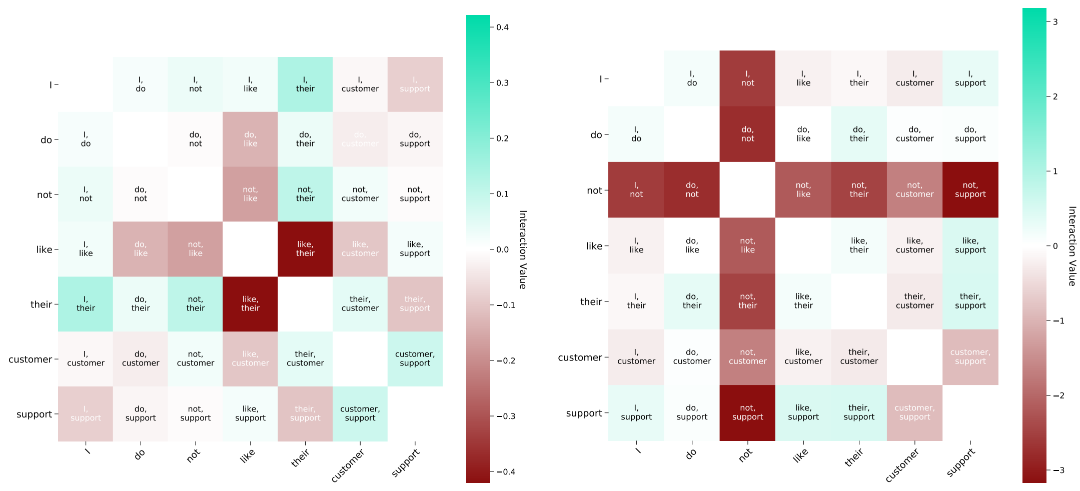
In this subsection, we briefly introduce the main frameworks used in this study; further details about all experiments, such as hyperparameters for each baseline and experiment, are provided in the supplementary materials. We selected official implementation for the LIME, SHAP, and IH baselines, and for the IG baseline, we employed the Captum library [43]. The graph structure was analyzed using the NetworkX library [44]. For all experiments we use a single NVIDIA 2080TI GPU with 12 GB of memory. For future comparison, we also open-source the code for the RLE framework for PyTorch [45] models and publish it online.

Method	IROF [39]
Random	0.179±0.18
IG [14]	0.211±0.23
LIME [13]	<u>0.421±0.19</u>
SHAP [18]	0.368±0.24
RLE (ours)	**0.434±0.23**

Table 3: A quantitative comparison of selected baselines on the fifty random images from ImageNet data set [24] given a pre-trained ResNet-50 model [16]. The top result is bold, whereas the second result is underlined.

5 Discussion and Future Work

Experimental results. In our challenging experiments with multiple data modalities, local explanations from the RLE framework show competitive performance against selected feature attribution baselines. Overall, our quantitative experimental results resemble similar image areas or words from highlighting by other state-of-the-art non relational explanation methods. In qualitative experiments, the proposed approach shows the best results on the IROF measure [39]. For more results, please refer to the supplementary material.



IH [23] RLE (ours)

Figure 5: A comparison of the relational local explanations from IH [23] and RLE methods for a sentence "`I do not like their customer support`", given a pre-trained `DistilBERT` model [16] for the sentiment analysis task. According to the IH method the most negative pair is "`like, their`", where our proposed approach shows the most negative word is "not" with two pairs "`not, support`" and "`do, not`". Appendix A presents more results.

Permutation step. One of the core steps of the proposed algorithm is the random permutation with a replacement - we refer to it as a *weak perturbation*. In comparison to other perturbation-based feature attribution methods which use a *strong perturbation*, e.g., perturb a data sample by adding random noise [13, 14] or removing parts of information [18], the RLE framework preserves the local attribution unchanged, only shuffling the global structure. Moreover, local details are more important than a global structure for deep neural network models, as shown for vision and textual modalities [28, 34, 46]. Another known issue related to the strong perturbation approaches for the local approximations, this type of perturbation leads to the out-of-the-distribution problem [47], which creates the vulnerability to adversarial attacks [15].

Evaluation measure for relational local explanations. Another point of the work's continuation with relational local explanations is the absence of evaluation technique. For future work, a trustworthy and plausibility measure is needed; this is challenging since there is no access to the ground truth. On the flip side, with an unambiguous measure, a possible strategy would involve direct optimization over it.

Ensembling of feature attributions. To improve the robustness of local and relational explanations, unsupervised ensemble techniques can be applied to the outputs of multiple runs of the explanation algorithm. By aggregating the outputs of multiple runs of the algorithm, we can effectively reduce the impact of any individual run that may have produced biased or inaccurate explanations. This approach has been shown to be effective at improving the robustness of explanations in a variety of contexts [48].

RLE limitations. The proposed approach does not currently support the quantification of higher-order interactions between features for the relational local explanations. A more complex graph-based representation can be utilized for this task for future work. Furthermore, patches of image data should have adequate local information, and the adequacy depends on the resolution of the images. In our experiments using the ImageNet data set - an image usually cropped into the $224 \times 224 \times 3$ format, we observe that we can operate with up to 36-49 patches (depending on the content of an image). Lastly, the current implementation of the RLE framework cannot be utilized on tabular modality since tabular data has no spatial relationships [49].

6 Conclusion

This paper introduces the RLE (Relational Local Explanations) method, a novel, model-agnostic approach for generating local explanations that addresses a common challenge in post hoc explanations, which is the interpretability of inter-variable relationships. This method provides a qualitative measure of how different feature attributions interact with each other, which is useful for various data types and problems where knowledge of the relationships between features is necessary. In addition, the RLE framework also offers standard feature attributions as local explanations, which provide insight into the specific contributions of each feature towards the final prediction made by a machine learning model. Through extensive visual and quantitative experiments, we demonstrate that the proposed RLE method performs comparably to state-of-the-art methods for generating comprehensive (relational) local explanations. These results suggest that RLE may be a valuable tool for practitioners seeking to understand and improve the performance of their machine learning models.

References

[1] Feiyu Xu, Hans Uszkoreit, Yangzhou Du, Wei Fan, Dongyan Zhao, and Jun Zhu. Explainable ai: A brief survey on history, research areas, approaches and challenges. In *CCF international conference on natural language processing and Chinese computing*, pages 563–574. Springer, 2019. (cited on p. 1, 4)

[2] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *International journal of computer vision*, 128(2):261–318, 2020. (cited on p. 1)

[3] Shervin Minaee, Yuri Y Boykov, Fatih Porikli, Antonio J Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2021. (cited on p. 1)

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. (cited on p. 1, 3, 7, 14, 15, 18, 19)

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. (cited on p. 1, 6)

[6] Wafaa S El-Kassas, Cherif R Salama, Ahmed A Rafea, and Hoda K Mohamed. Automatic text summarization: A comprehensive survey. *Expert Systems with Applications*, 165:113679, 2021. (cited on p. 1)

[7] Marouane Birjali, Mohammed Kasri, and Abderrahim Beni-Hssane. A comprehensive survey on sentiment analysis: Approaches, challenges and trends. *Knowledge-Based Systems*, 226:107134, 2021. (cited on p. 1)

[8] Iam Palatnik De Sousa, Marley Maria Bernardes Rebuzzi Vellasco, and Eduardo Costa Da Silva. Local interpretable model-agnostic explanations for classification of lymph node metastases. *Sensors (Basel, Switzerland)*, 19(13), 2019. (cited on p. 1)

[9] Yu Zhang, Peter Tiňo, Aleš Leonardis, and Ke Tang. A survey on neural network interpretability. *arXiv preprint arXiv:2012.14261*, 2020. (cited on p. 1)

[10] Council of the European Union European Parliament. Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016. *Official Journal of the European Union*, 2016. (cited on p. 1)

[11] CA OAG. Ccpa regulations: Final regulation text. *Office of the Attorney General, California Department of Justice*, 2021. (cited on p. 1)

[12] Umang Bhatt, Alice Xiang, Shubham Sharma, Adrian Weller, Ankur Taly, Yunhan Jia, Joydeep Ghosh, Ruchir Puri, José MF Moura, and Peter Eckersley. Explainable machine learning in deployment. In *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pages 648–657, 2020. (cited on p. 1)

[13] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should I trust you? explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016. (cited on p. 1, 3, 4, 5, 7, 8, 9, 13, 15, 16)

[14] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017. (cited on p. 1, 3, 4, 7, 8, 9, 13, 15, 16)

[15] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. Fooling lime and shap: Adversarial attacks on post hoc explanation methods. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 180–186, 2020. (cited on p. 1, 5, 9)

[16] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019. (cited on p. 2, 6, 7, 8, 9, 14, 16, 17)

[17] Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. Explainable ai for trees: From local explanations to global understanding. *arXiv preprint arXiv:1905.04610*, 2019. (cited on p. 3)

[18] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc., 2017. (cited on p. 3, 4, 5, 7, 8, 9, 13, 15, 16)

[19] Tianyu Cui, Pekka Marttinen, and Samuel Kaski. Learning pairwise interactions with bayesian neural networks. *arXiv preprint arXiv:1901.08361*, 2019. (cited on p. 3)

[20] Peyton Greenside, Tyler Shimko, Polly Fordyce, and Anshul Kundaje. Discovering epistatic feature interactions from neural network models of regulatory dna sequences. *Bioinformatics*, 34(17):i629–i637, 2018. (cited on p. 3)

[21] Chandan Singh, W James Murdoch, and Bin Yu. Hierarchical interpretations for neural network predictions. *arXiv preprint arXiv:1806.05337*, 2018. (cited on p. 3)

[22] W James Murdoch, Peter J Liu, and Bin Yu. Beyond word importance: Contextual decomposition to extract interactions from lstms. *arXiv preprint arXiv:1801.05453*, 2018. (cited on p. 3)

[23] Joseph D Janizek, Pascal Sturmfels, and Su-In Lee. Explaining explanations: Axiomatic feature interactions for deep networks. *Journal of Machine Learning Research*, 22(104):1–54, 2021. (cited on p. 3, 4, 8, 9, 13, 14, 16, 17)

[24] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. (cited on p. 3, 7, 8, 13, 15, 18, 19)

[25] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018. (cited on p. 4)

[26] Damien Garreau and Ulrike Luxburg. Explaining the explainer: A first theoretical analysis of lime. In *International Conference on Artificial Intelligence and Statistics*, pages 1287–1296. PMLR, 2020. (cited on p. 4)

[27] Jürgen Dieber and Sabrina Kirrane. Why model why? assessing the strengths and limitations of lime. *arXiv preprint arXiv:2012.00093*, 2020. (cited on p. 4)

[28] Yue Chen, Yalong Bai, Wei Zhang, and Tao Mei. Destruction and construction learning for fine-grained image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5157–5166, 2019. (cited on p. 6, 9)

[29] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6707–6717, 2020. (cited on p. 6)

[30] Saeid Asgari Taghanaki, Mohammad Havaei, Alex Lamb, Aditya Sanghi, Ara Danielyan, and Tonya Custis. Jigsaw-vae: Towards balancing features in variational autoencoders. *arXiv preprint arXiv:2005.05496*, 2020. (cited on p. 6)

[31] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. (cited on p. 6)

[32] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. (cited on p. 6)

[33] Alberto Sanfeliu, René Alquézar, J Andrade, Joan Climent, Francesc Serratosa, and J Vergés. Graph-based representations and techniques for image processing and image analysis. *Pattern recognition*, 35(3):639–650, 2002. (cited on p. 6)

[34] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019. (cited on p. 6, 9)

[35] Vadim Borisov, Kathrin Seßler, Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. Language models are realistic tabular data generators. *arXiv preprint arXiv:2210.06280*, 2022. (cited on p. 6)

[36] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018. (cited on p. 6)

[37] Chaitanya Joshi. Transformers are graph neural networks. *The Gradient*, 2020. (cited on p. 6)

[38] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45, 2020. (cited on p. 7, 14)

[39] Laura Rieger and Lars Kai Hansen. Irof: a low resource evaluation metric for explanation methods. *arXiv preprint arXiv:2003.08747*, 2020. (cited on p. 8, 14)

[40] Umang Bhatt, Adrian Weller, and José MF Moura. Evaluating and aggregating feature-based model explanations. *arXiv preprint arXiv:2005.00631*, 2020. (cited on p. 8)

[41] Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. Evaluating feature importance estimates. *arXiv preprint arXiv:1806.10758*, 2018. (cited on p. 8)

[42] Yao Rong, Tobias Leemann, Vadim Borisov, Gjergji Kasneci, and Enkelejda Kasneci. A consistent and efficient evaluation strategy for attribution methods. In *International Conference on Machine Learning*, pages 18770–18795. PMLR, 2022. (cited on p. 8)

[43] Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, and Orion Reblitz-Richardson. Captum: A unified and generic model interpretability library for pytorch, 2020. (cited on p. 8, 13)

[44] Aric Hagberg, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008. (cited on p. 8)

[45] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. (cited on p. 8, 14)

[46] Zilong Huang, Youcheng Ben, Guozhong Luo, Pei Cheng, Gang Yu, and Bin Fu. Shuffle transformer: Rethinking spatial shuffle for vision transformer. *arXiv preprint arXiv:2106.03650*, 2021. (cited on p. 9)

[47] Peter Hase, Harry Xie, and Mohit Bansal. The out-of-distribution problem in explainability and search methods for feature importance explanations. *Advances in Neural Information Processing Systems*, 34, 2021. (cited on p. 9)

[48] Vadim Borisov, Johannes Meier, Johan van den Heuvel, Hamed Jalali, and Gjergji Kasneci. A robust unsupervised ensemble of feature-based explanations using restricted boltzmann machines. *arXiv preprint arXiv:2111.07379*, 2021. (cited on p. 9)

[49] Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. Deep neural networks and tabular data: A survey. *arXiv preprint arXiv:2110.01889*, 2021. (cited on p. 9)

[50] Naman Bansal, Chirag Agarwal, and Anh Nguyen. Sam: The sensitivity of attribution methods to hyperparameters. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pages 8673–8683, 2020. (cited on p. 13)

[51] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282, 2012. (cited on p. 14)

A Additional Experiments

This section present experimental results on visual (Table 4) and textual (Table 5) data. Additionally, we show relational local explanations for several visual and text samples in Figures 6, 7, 8, 9, 10, 11, 12. We compare results from the proposed algorithm to the baselines of this study: IG [14], LIME [13], SHAP [18], and IH [23].

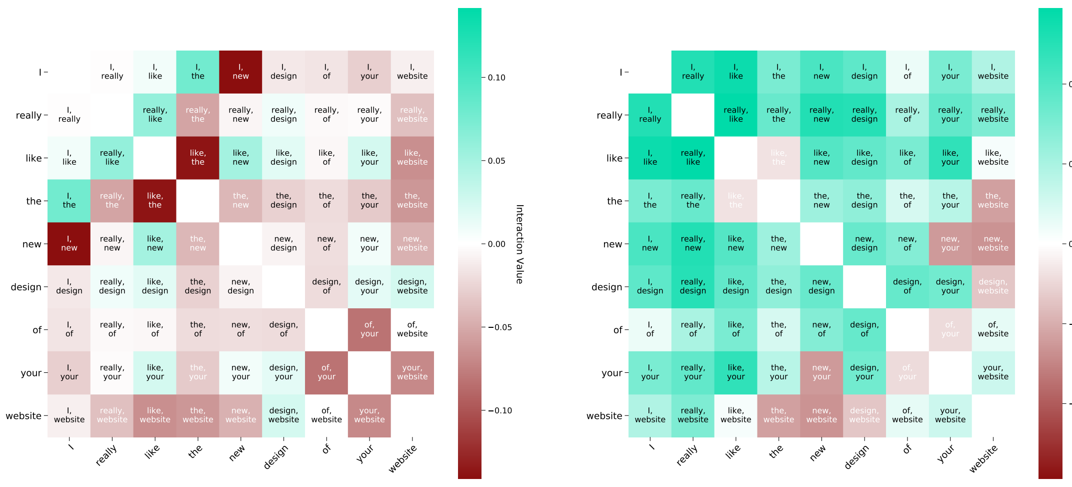
B Further Reproducibility Details

Hyperparameters. We select similar hyperparameters for each baseline to have a fair evaluation; for image data, the number of perturbations (auxiliary samples) n is set to 5000, and for textual, we set n to 2000. In our experiments, we observe that a higher number of perturbation steps leads to better quality local explanations. This was also observed in [50]. The rest of the hyperparameters default to a selected package.

RLE plotting function. For the relational local explanation visualization we apply a plotting function from the IH [23] official implementation.[2] From the open-source library Captum [43], we utilize plotting function for visualization feature attribution maps on visual data.[3]

Data sets. For image data, we utilize samples from ImageNet data set [24] provided by the official python package of the SHAP algorithm [18].[4]

[2]https://github.com/suinleelab/path_explain
[3]https://captum.ai/api/utilities.html
[4]https://shap.readthedocs.io/en/latest/generated/shap.datasets.imagenet50.html



IH [23] RLE (ours)

Figure 6: A comparison of the relational local explanations from IH [23] and RLE methods for a sentence “`I really like the new design of your website`”, given a pre-trained DistilBERT model [16] for the sentiment analysis task.

Pre-trained models. In this work, we employ the pre-trained ResNet-50 model [4] from the *torchvision* package [45].[5]. We utilize the pre-trained DistilBERT model [16] from the *HugginFace* library [38].[6]

For even better reproducibility, we also report the used package versions in `requirements.txt` file. It can be found in the corresponding code repository of the RLE algorithm.

C The IROF Measure

Choice of the quantitative measure. In our work, we select the IROF framework [39], since it allows for an efficient and fairly evaluation of feature attribution methods for the visual data. In comparison to popular approaches for single-pixel-based evaluation of local explanations (e.g., DAUC, IAUC), the chosen evaluation framework uses the super-pixel approach. Since the influence of a single pixel is minimal, the unsupervised grouping of a pixel into local regions allows us for a more fair comparison of the feature attribution methods.

The IROF approach has several steps: First, the image is divided into coherent segments and bypasses the input features’ inter-dependency. According to the creators of the IROF measure, we use the SLIC method for unsupervised image segmentation [51].

Formally, the IROF measure is defined as follows:

$$\mathrm{IROF}(e_j) = \frac{1}{N}\sum_{n=1}^{N} \mathrm{AOC}\left(\frac{f(x_n^0)_y}{f(x_n^0)_y}\right)_{l=0}^{L} \tag{4}$$

where e_j is a local feature attribution map, N is the number of super-pixels, x^0 is an image to explain, f is a black-box model, y is a target, and L represents the class score based on how many segments of the image were removed. Also, the proposed measure utilized the area over the curve (AOC) function. The higher the IROF score, the more plausible the local explanations, i.e., the more information was collected.

[5] https://pytorch.org/vision/stable/models.html

[6] https://huggingface.co/distilbert-base-uncased-finetuned-sst-2-english

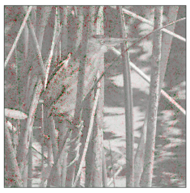
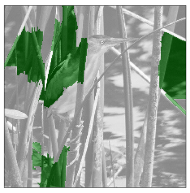
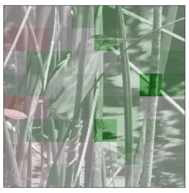
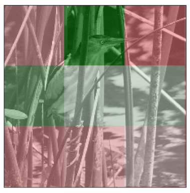
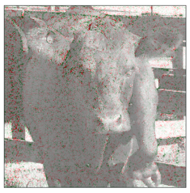
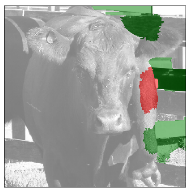
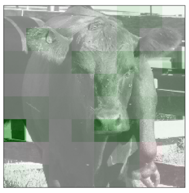
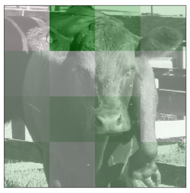
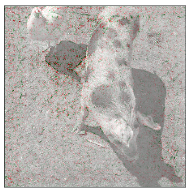
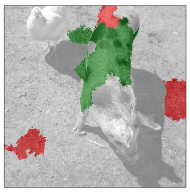
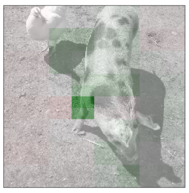
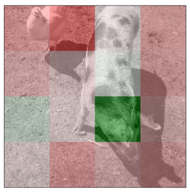
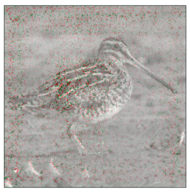
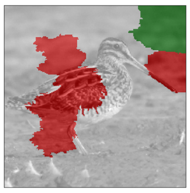
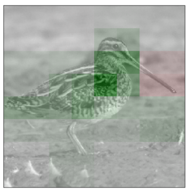
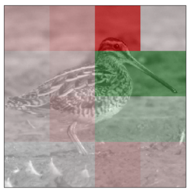
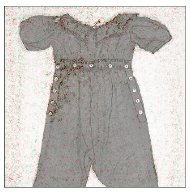
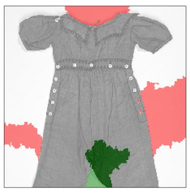
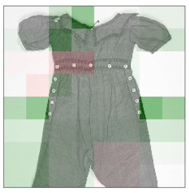
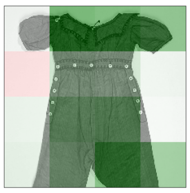
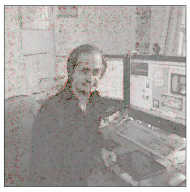
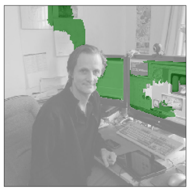
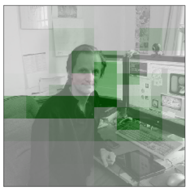
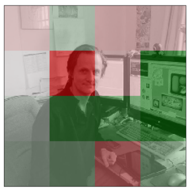
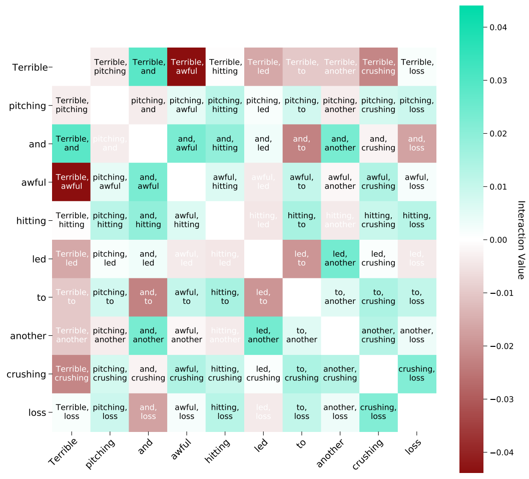
Original	IG [14]	LIME [13]	SHAP [18]	RLE (Ours)
				
				
				
				
				
				

Table 4: A Comparison of different state-of-the-art local explanation methods for a pre-trained ResNet-50 model [4] given random samples from the ImageNet data set [24]. Name of the original classes *according* to the selected model (from top to bottom): `bittern`, `Indian elephant`, `hog`, `dowitcher`, `cardigan`, and `desktop computer`. The explanations from the RLE method are less noisy and rank almost all parts of the image with either positive (green) or negative (red) influence.

Method	Local Explanation
IG [14]	The new design is awful
LIME [13]	The new design is awful
SHAP [18]	The new design is awful
IH [23]	The new design is awful
RLE (ours)	The new design is awful
IG [14]	I love you and I hate you
LIME [13]	I love you and I hate you
SHAP [18]	I love you and I hate you
IH [23]	I love you and I hate you
RLE (ours)	I love you and I hate you
IG [14]	I'm not sure if I like the new design
LIME [13]	I'm not sure if I like the new design
SHAP [18]	I'm not sure if I like the new design
IH [23]	I'm not sure if I like the new design
RLE (ours)	I'm not sure if I like the new design
IG [14]	I really like the new design of your website
LIME [13]	I really like the new design of your website
SHAP [18]	I really like the new design of your website
IH [23]	I really like the new design of your website
RLE (ours)	I really like the new design of your website
IG [14]	The bed was super comfy. The chair wasn't bad, either
LIME [13]	The bed was super comfy. The chair wasn't bad, either
SHAP [18]	The bed was super comfy. The chair wasn't bad, either
IH [23]	The bed was super comfy. The chair wasn't bad, either
RLE (ours)	The bed was super comfy. The chair wasn't bad, either
IG [14]	Terrible pitching and awful hitting led to another crushing loss
LIME [13]	Terrible pitching and awful hitting led to another crushing loss
SHAP [18]	Terrible pitching and awful hitting led to another crushing loss
IH [23]	Terrible pitching and awful hitting led to another crushing loss
RLE (ours)	Terrible pitching and awful hitting led to another crushing loss

Table 5: A comparison of state-of-art feature attribution approaches to the presented RLE algorithm given a pre-trained DistilBERT model [16] for the sentiment analysis task. We highlight the most important words according to each feature attribution method, where the green and red colors indicate the positive and negative impact, respectively.


IH [23]

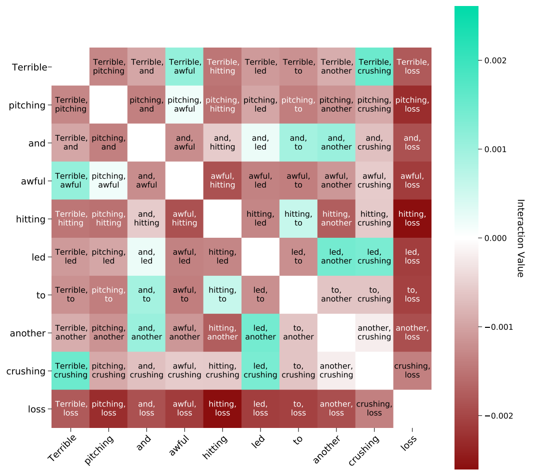
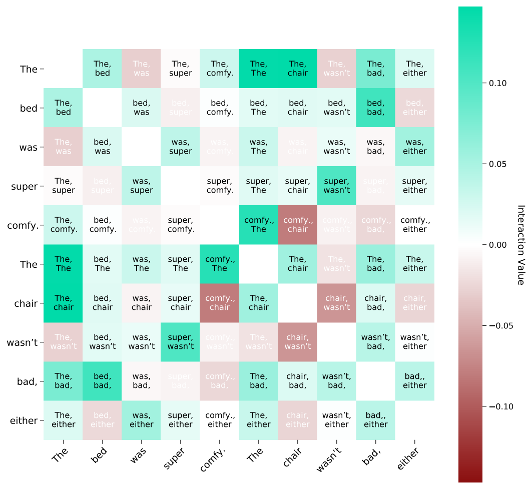

RLE (ours)

Figure 7: A comparison of the relational local explanations from IH [23] and RLE methods for a sentence "`Terrible pitching and awful hitting led to another crushing loss`", given a pre-trained DistilBERT model [16] for the sentiment analysis task.


IH [23]

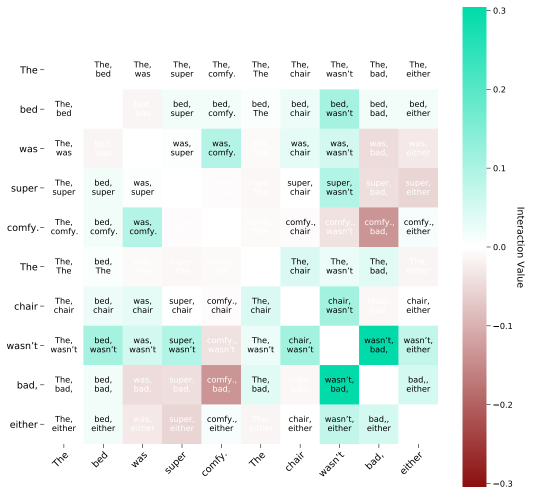

RLE (ours)

Figure 8: A comparison of the relational local explanations from IH [23] and RLE methods for a sentence "`The bed was super comfy.  The chair wasn't bad, either`", given a pre-trained DistilBERT model [16] for the sentiment analysis task.

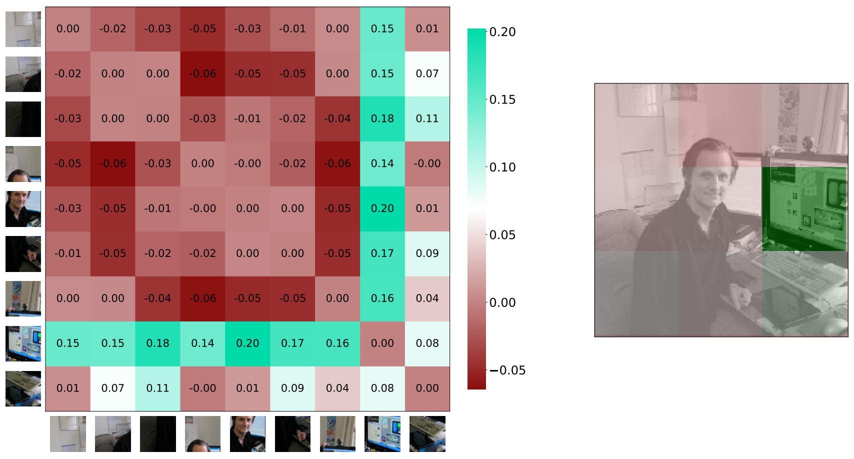


Figure 9: An example of the relational local explanation (*left*) and standard local explanation (*right*) for visual data from the proposed RLE framework where green color indicates positive influence, and red negative. For the task we select a pre-trained ResNet-50 model [4] and an image with a class `desktop computer` from the ImageNet data set [24], e.g., to uncover a combination of patches that is the most important to a model.

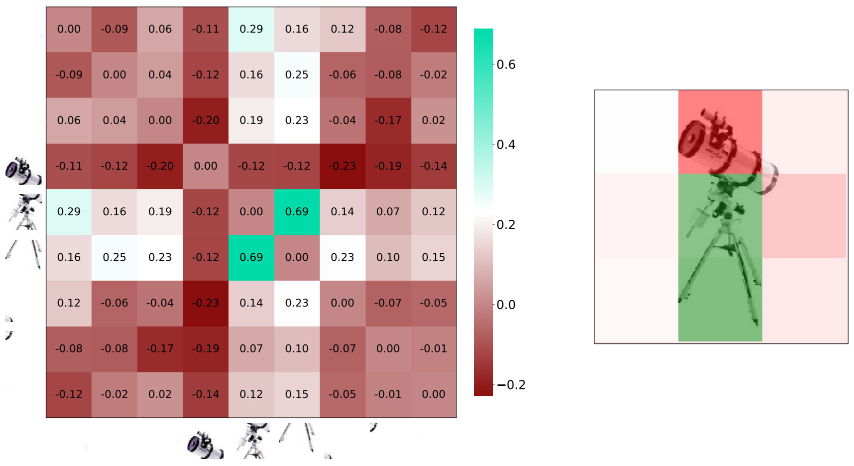


Figure 10: An example of the relational local explanation (*left*) and standard local explanation (*right*) for visual data from the proposed RLE framework where green color indicates positive influence, and red negative. For the task we select a pre-trained ResNet-50 model [4] and an image with a class `tripod` from the ImageNet data set [24], e.g., to uncover a combination of patches that is the most important to a model.

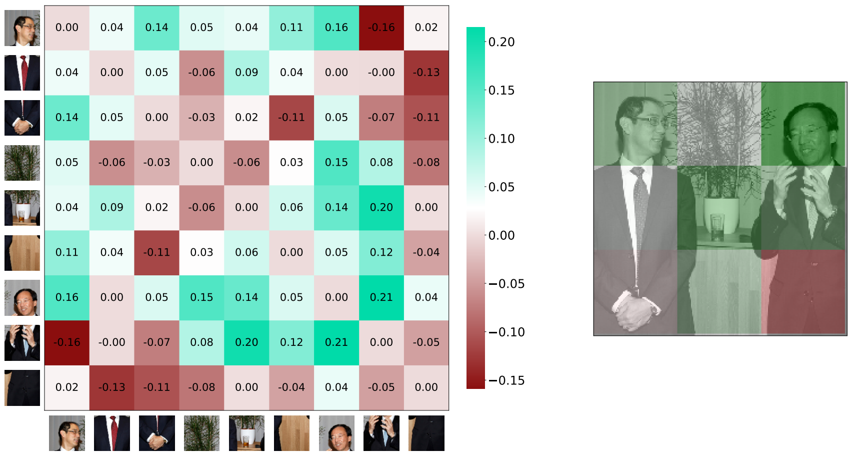


Figure 11: An example of the relational local explanation (*left*) and standard local explanation (*right*) for visual data from the proposed RLE framework where green color indicates positive influence, and red negative. For the task we select a pre-trained ResNet-50 model [4] and an image with a class `groom` from the ImageNet data set [24], e.g., to uncover a combination of patches that is the most important to a model.

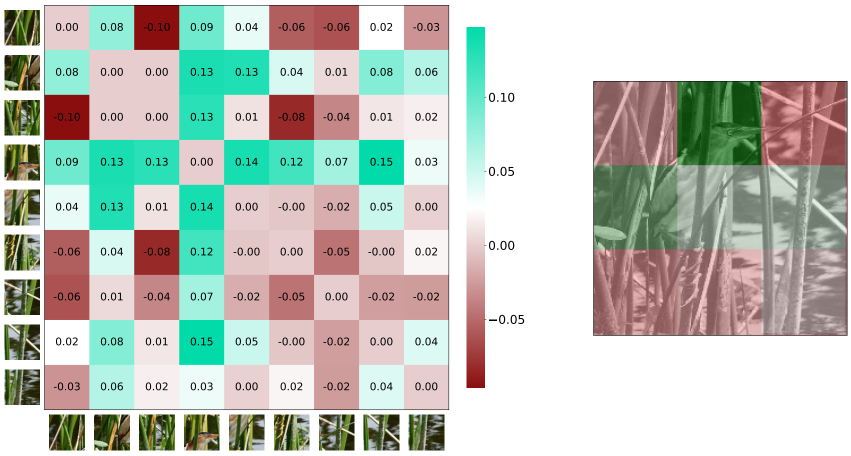


Figure 12: An example of the relational local explanation (*left*) and standard local explanation (*right*) for visual data from the proposed RLE framework where green color indicates positive influence, and red negative. For the task we select a pre-trained ResNet-50 model [4] and an image with a class `bittern` from the ImageNet data set [24], e.g., to uncover a combination of patches that is the most important to a model.

A.3 Applied Manuscripts

A.3.1 Robust Cognitive Load Detection from Wrist-Band Sensors

Publication: Published in the computers in human behavior reports journal, 2021.

Contribution: I developed all major parts of the explainability framework and the implementations of the relational local explanation framework. I also performed the experiments and wrote most parts of the paper. Gjergji Kasneci and Enkelejda Kasneci contributed to the paper by challenging and improving ideas, formalizations, and the analysis of results. All co-authors helped revise the final manuscript.

Contents lists available at ScienceDirect

Computers in Human Behavior Reports

journal homepage: www.journals.elsevier.com/computers-in-human-behavior-reports

Robust cognitive load detection from wrist-band sensors

Vadim Borisov [*], Enkelejda Kasneci, Gjergji Kasneci

The University of Tübingen, Germany

ARTICLE INFO

Keywords:
Cognitive load detection
Human-computer interaction
Machine learning
Ensemble methods

ABSTRACT

In recent years, the detection of cognitive load has received a lot of attention. Understanding the circumstances in which cognitive load occurs and reliably predicting such occurrences, offers the potential for considerable advances in the field of Human-Computer Interaction (HCI). Numerous HCI applications, ranging from medical and health-related solutions to (smart) automotive environments, would directly benefit from the reliable detection of cognitive load. However, this task still remains highly challenging. We present a machine learning (ML) approach based on ensemble learning for robust cognitive load classification. The features used by the proposed solution are generated from the interpretation of physiological measurements (e.g., heart rate, r-r interval, skin temperature, and skin response) from a wearable device. Hence, our approach consists of two steps: (1) transforming the original data into discriminative features and (2) training an ensemble model to accurately and robustly predict cognitive load. The empirical results confirm that our method has a superior performance compared to various state-of-the-art baselines on the original and transformed data. Moreover, in the open-data CogLoad@UbiComp 2020 Competition, the proposed approach achieved the best results among 17 competing approaches and outperformed all participating competitors by a considerable margin.

1. Introduction

The degree to which our cognitive resources, e.g., attention, working memory, decision making, or task-related knowledge, are currently used is commonly referred to as cognitive load (Sweller, 2011). With emerging novel multimedia technologies, cognition-aware computing, and human-centered systems that aim to automatically adapt to the user's cognitive state, predicting and quantifying cognitive load can be beneficial in various applications of Human-Computer Interaction (HCI). More specifically, the ability to estimate the proximal zone of a user, where stress, frustration (as typically resulting from cognitive overload), and boredom (originating from low levels of cognitive load) can be avoided, promises to open new avenues towards the development of truly intelligent user-centered systems and enhanced user experience. For example, intelligent tutoring systems could have crucially aided learning and teaching in the current COVID-19 epidemiological situation. Beyond the learning context, in entertainment-related applications such as gaming, the online assessment of a user's cognitive load could contribute significantly to an enhanced user experience. In various medical applications, the ability to detect the cognitive overload of medical experts could help in the development of appropriate supporting measures and systems. Considered a highly important measure towards a better understanding of human cognition and performance, the measurement and prediction of cognitive load has been the focus of research works for more than three decades.

A variety of approaches to cognitive load have been explored during the past decades (Kramer, 1990; Sweller, 2011), ranging from questionnaire-based techniques such as the NASA TLX (Hart & Staveland, 1988) self-report to advanced methods based on Deep Neural Networks on image data (Fridman et al., 2018). The main limitation of self-reports, however, is the subjective nature of the user responses, which hinders the identification of ground truth labels. Furthermore, it has been shown that such questionnaires may induce additional load to the user (Abdelrahman et al., 2017) and are not applicable to online settings where a person's cognitive load has to be estimated during task performance. Therefore, with increasing technological possibilities for user monitoring (e.g., through electroencephalography, eye-tracking technology, camera-based user monitoring, or galvanic skin response and heart rate sensors), estimating cognitive load based on physiological or image data of a user has progressively gained research focus.

This work aims to achieve the automated detection of cognitive load on physiological user data sensed *non-invasively* from a wearable using machine learning techniques. More specifically, we present the 1st place solution from the CogLoad@UbiComp 2020 Competition, which

* Corresponding author.
E-mail addresses: vadim.borisov@uni-tuebingen.de (V. Borisov), enkelejda.kasneci@uni-tuebingen.de (E. Kasneci), gjergji.kasneci@uni-tuebingen.de (G. Kasneci).

https://doi.org/10.1016/j.chbr.2021.100116
Received 10 February 2021; Received in revised form 22 June 2021; Accepted 23 June 2021
Available online 29 June 2021

addressed cognitive load detection from low-cost wrist-band sensors (Microsoft Band 2), i.e. measuring galvanic skin response (GSR), skin temperature (ST), heart rate (HR), and heart rate variability (RR intervals). Our method consists of two steps; first, data is transformed from temporal to static data, allowing all standard ML algorithms to be used. The second step is ensemble learning using decision tree-based models, which is proved to be robust on various tasks, especially on data.

In summary, the contributions of this work are multi-fold:

- We present the winning solution of the CogLoad@UbiComp 2020 Competition. The proposed method outperformed all other state-of-the-art approaches from competitors in terms of predictive performance on unseen test data by a considerable margin.
- We demonstrate how simple yet effective data transformation techniques can improve the state-of-the-art machine learning approaches, thereby advancing the current state-of-the-art in the area of cognitive load detection based on wearable sensors.
- We provide a comprehensive comparison of the proposed ensemble approach with other state-of-the-art machine learning methods on the original and transformed data.
- An open-source implementation of our approach can be found here: https://www.github.com/unnir/CogLoad_UbiComp2020.

The remainder of this article is organized as follows. In Section 2, we first provide an overview of related work on cognitive load detection from physiological data, and especially on methods approaching the CogLoad@UbiComp 2020 Challenge. The cognitive load data set from this challenge is described in detail in Section 3. Section 4 presents our feature transformation methods, followed by our model description for accurate cognitive load detection in Section 5. The experimental evaluation and performance results of our model are presented and discussed in Section 6. In Section 7, we discuss the limitation of our approach with future steps. Section 8 concludes this work.

2. Related work

Beyond questionnaires and self-reports, user monitoring holds enormous potential for the robust measurement of a person's cognitive load in an online fashion, i.e. during task performance. This information can, in turn, be employed by the systems with which the user is interacting and, thus, become beneficial for the user's performance through adaptation or additional supportive measures. Various approaches have therefore been proposed during in recent years to tackle the detection of cognitive load and will be discussed in the following along with specific solutions and implementations on the same data set of the CogLoad@UbiComp2020 Challenge.

2.1. Detecting cognitive load from brain activity

Common neuro-imaging techniques that have been used to detect cognitive load are electroencephalography (EEG), e.g., as in (Friedman et al., 2019; Mills et al., 2017), near-infrared spectroscopy (NIRS, e.g., in (Grubov et al., 2020; Keshmiri et al., 2017)), or functional magnetic resonance imaging (fMRI, as for example from (Mäki-Marttunen et al., 2019)). Although brain imaging techniques promise to deliver highly accurate detection and prediction of cognitive load, the technology has not yet reached the stage of ubiquity and low-cost availability. These techniques are therefore not yet applicable to use-cases outside the laboratory.

2.2. Detecting cognitive load from eye movements and pupil information

With recent advances in eye-tracking technology, a non-intrusive way to infer information about a person's cognitive state is becoming available. In recent years, different features of the eye and of eye movements have been investigated as predictors of cognitive load. For example (Chen et al., 2011; He & McCarley, 2010; Inamdar & Pomplun, 2003; Van Orden et al., 2001; Wang et al., 2014), associated longer fixations with more effort and thus with higher levels of cognitive load. However, other streams of related work have indicated the opposite, i.e., by relating longer fixation to lower cognitive effort, e.g., (Amadieu et al., 2009; Van Gog et al., 2005). These results, however, might be influenced by processing difficult or visually challenging stimuli (Rayner, 1998). In addition, a few recent articles have looked at the relationship between smooth pursuits and cognitive load (Kosch et al., 2018; Stubbs et al., 2018), and reported a high predictive power of features extracted from smooth pursuits on cognitive load. In the past few years, enabled through high-speed eye-tracking devices, a research line in this area has investigated the relationship between microsaccades and cognitive load. Microsaccades describe small, involuntary eye movements which occur during a fixation period and are assumed to be highly associated with cognitive and visual load. More specifically, tasks that induce high visual load were found to cause an increase of the frequency of microsaccades (Benedetto et al., 2011), while auditory or arithmetic tasks have been found to have the opposite effect, i.e., reduce their frequency (Gao et al., 2015; Krejtz et al., 2018; Siegenthaler et al., 2014). In addition to eye-movement characteristics, blinks and their frequency have also been investigated regarding their relation to visual or cognitive load (Bristow et al., 2005; Fukuda et al., 2005). More specifically, it has been reported that higher cognitive load induces more frequent blinking (Chen & Epps, 2014; Hogervorst et al., 2014).

Another line of research in this area focuses on predictive features of cognitive load derived from the eye pupil and its oscillations. It is well understood that high cognitive load causes characteristic patterns of pupil dilation (Beatty et al., 2000; Kramer, 1991), which is also known as the task-evoked pupillary response. It has been further shown that this effect even persists within a task, between tasks, and between individuals, concluding that there is a consistent influence of cognitive load on the pupil diameter (Kahneman, 1973). Since changing illumination conditions (e.g., environmental lightning or lightning changes of the visual stimulus itself) also affect pupillary response, several research articles have focused on generalizable approaches, e.g., (Appel et al., 2018, 2019; Duchowski et al., 2018, pp. 1–13; Faure et al., 2016; Kun et al., 2013; Marshall, 2000, 2007; Palinko et al., 2010; Pfleging et al., 2016).

2.3. Image-based approaches to cognitive load detection

Recent approaches have proposed (deep-learning) frameworks for cognitive load detection based on image data (Fridman et al., 2018; Rafiqi et al., 2015) or thermal images (Abdelrahman et al., 2017). The latter group of methods aims to automatically detect changes in skin temperature, respiration, or heart rate related to changing levels of cognitive load based on thermal images. Although image-based technology is, in general (primarily due to its non-intrusive nature), highly interesting and relevant, further research is required to determine the generalizability of such approaches across tasks and subjects.

2.4. Physiological signals for cognitive load estimation: galvanic skin response (GSR), heart rate (HR), and heart rate variability (HRV)

Physiological sensors measuring GSR, HR, and HRV[1] are meanwhile available at low cost and large scale. Thus, these signals can be employed for user monitoring in a variety of applications. As a measure of skin conductivity, GSR is considered a strong indicator of stress and cognitive load and, as such, is used in various approaches to detect cognitive load. More specifically, various studies have reported an increase in GSR with

[1] Note that HRV corresponds to R-R intervals. We use the short form RR in the following sections, which is in accordance with the terminology used in the data set description of the CogLoad@UbiComp 2020 Challenge.

increasing cognitive load, e.g. (Kasneci et al., 2017; Nourbakhsh et al., 2012, 2017; Shi et al., 2007). Cardiac measures such as HR and HRV have also been successfully employed in many related studies on cognitive load detection, e.g. (Gjoreski et al., 2018; Hughes et al., 2019; Kübler et al., 2014; McDuff et al., 2014; Mehler et al., 2011; Wang & Guo, 2019).

Apart from the above-mentioned lines of related research, various approaches have proposed multimodal methods of cognitive load detection to increase the accuracy of predictions (Debie et al., 2021). Prabhakar et al. (2020) suggested the estimation of cognitive load from eye-movement and pupil dilation parameters. In four independent studies with 123 participants, Sharma et al. (2020) analysed the assessment of cognitive load through physiological responses and facial expressions, a method that works well for recognizing successful perception, at least by aggregating physiological and eye-tracking signals (Kasneci et al., 2017). Finally, in the aviation industry, the estimation of the cognitive load plays a crucial role and has therefore been analysed in various studies, e.g., (Babu et al., 2019; Di Nocera et al., 2007; Wilson et al., 1994).

Our work specifically addresses the applicability of machine learning models to the prediction of the cognitive load of a user from physiological data collected from smart wearables, since these sensors are not only non-intrusive and affordable, but also convenient to use. Hence, in the following, we will briefly discuss the state-of-the-art in this area as well as competing approaches from the same challenge.

2.5. *Cognitive load estimation from wearable sensors*

Setz et al. (2010) introduced a machine learning approach in 2009 based on six classifiers to distinguish stress from cognitive load in an office environment based on data from 33 subjects in a laboratory intervention study. Their methods achieved an accuracy of 82.8%, thus exhibiting promising steps towards cognitive load detection based on low-cost smart devices. Huang et al. (2018) showed that physiological signals from wearable devices could be used to analyze psychological factors that can help with disease prevention. In another study, Schaule et al. (2018) introduced a system coined COLLINS (COgnitive Load CLassification to prevent INterruptionS), which utilized a smart wrist-band device for cognitive load estimation. This approach showed that the cognitive load could be estimated using sensors from a smart device. In an evaluation with ten subjects the authors compared three machine learning algorithms - SVM, Random Forest, and Naive Bayes (SVM and Random Forest are selected as baselines for our experiments) and reported an accuracy between 66% and 86% for individual participants. For the general classification task, COLLINS achieved an accuracy between 32% and 36% in a ten-fold cross-validation. The work from (Gjoreski et al., 2018), based on 25 volunteers, utilized sensor information from a simple wearable device in order to measure the cognitive load. In parallel, they collected physiological data with a device, extracted features, and then constructed machine learning models for cognitive load prediction. Although the final accuracy of the statistical model was only 51%, the work confirmed that it is possible to estimate cognitive load using a wearable device. The data collected by the authors was later released to the research community as a challenge and served as a data foundation of our work.

2.6. *Related approaches from the CogLoad@UbiComp 2020 challenge*

The CogLoad@UbiComp 2020 Competition, along with an open data set, was advertised by the 5th International Workshop on Smart & Ambient Notification and Attention Management (UbiTention 2020) at UbiComp 2020 (Li & De Cock, 2020) (team Lynx from the University of Washington). used different combinations of feature engineering steps (e.g., Fast Fourier Transforms, Sliding Mean Filter) in combination with machine learning models (e.g., Logistic Regression, (Boosted) Decision Trees, Random Forests, and Support Vector Machines) to approach the problem. Their best processing pipelines yielded an accuracy of 63% on the data set, which is in line with previous work on smartwatch data (Gjoreski et al., 2020). In another work on this data set, Salfiger (Salfinger, 2020) investigated the applicability of deep learning approaches for cognitive load monitoring. More specifically, the author evaluated different configurations of Recurrent Neural Networks (Schmidhuber, 2015), namely Gated Recurrent Units (GRUs) and RNNs using Long short-term memory (LSTM) cells, and found that architectures based on GRUs achieve the best performance. A limitation of this approach is however the complexity of the models which have a tendency to overfit with respect to the small size of the data set from the challenge, which, in addition, is characterized by considerable between-subject variance and subject-related bias (Salfinger, 2020). The team HCM-lab from the University of Augsburg, Germany, got second place in the competition by using a deep learning-based approach. First, they trained an autoencoder on original data. Second, they utilized the encoder only as an input block and used a three-layer artificial neural network on top of it. A team from the VTT Technical Research Centre of Finland (Tervonen et al., 2021) used a support-vector machine (SVM) based (Boser et al., 1992) approach with the Bayesian optimization step. Their best mode received the 3rd place in the CogLoad@UbiComp 2020 competition.

3. The data set

The method we propose in this work was developed and evaluated in the context of the CogLoad challenge (van Berkel et al.) using the CogLoad data set (Gjoreski et al., 2020) provided by UbiComp 2020 (international joint conference on pervasive and ubiquitous computing, 2020), the leading venue in the area of ubiquitous and pervasive computing. The data set is freely available online.[2]

The data set contains four different physiological measurements, which were recorded by a Microsoft Band 2 wrist-band from 23 participants performing six psychological tasks on a PC with varying levels of difficulty, as well as measurements recorded while the participants were in a rest state. Participants' mean age was 29.51 (standard deviation is 10.10). The right was the dominant hand of 22 participants, while 1 participant was left-handed. All participants had the wrist-band device strapped to their left hand. In the conducted trials, the participants solved cognitive tasks of varying difficulty. The experiments were conducted in a quiet and normal-temperature room with one participant at a time. The experimental scenario consisted of two parts. Part 1 was devoted to estimating the participants' cognitive capacity. For assessing the participants' cognitive capacity, the participants solved two N-back tasks (Schmiedek et al., 2014), i.e., 2-back and 3-back tasks, with a 3-min rest after each of them. In Part 2, the participants were presented with six primary tasks. For each task, three variations of a randomly selected primary cognitive-load task were presented to the participant. The variations differed in complexity (easy, medium, and difficult). More information on participants and task are provided in (Gjoreski et al., 2020).

For each participant, 50% of the samples correspond to the cognitive load state, the other 50% to resting. In the data set, the target variable is represented by a binary value, i.e., a ‘1’ represents a low cognitive load, i.e., resting, and a ‘0’ represents a high cognitive load.

The physiological measurements include Galvanic skin response (GSR), heart rate (HR), R-R intervals (RR), and skin temperature (ST). All these measurements were sampled at a sampling rate of 1Hz. From the 23 participants, the recorded measurements of 5 participants were used for testing and the measurements of the remaining 18 participants for training. The training and test data were generated using time windows of 30 s.

The sensor files in the training data (GSR, HR, RR, and ST) contain 632 lines x 30 columns, corresponding to 632 instances each containing 30 samples (i.e., generated within 30 s at the sampling rate 1Hz). The training instances are randomly permuted. The sensor files in the test

[2] CogLoad@UbiComp Data set: https://www.ubittention.org/2020/.

data (GSR, HR, RR, and ST) contain 193 lines x 30 columns, corresponding to 193 instances each containing 30 samples (30 s at the sampling rate 1 Hz). The test instances are also randomly permuted.

Note, that apart from sampling and resampling, no additional preprocessing steps (e.g., a fast Fourier transform (FFT) filtering) were used. Thus, the data is raw as provided by the Microsoft Band 2. As an example, the data for one of the participants is visualized in Fig. 1.

4. Transformation of the raw data into discriminative features

Each instance from the raw training and test data set can be represented as a vector of length 120, i.e., as a concatenation of the vectors of length 30 from the single physiological signals recorded in the 30-s window. More specifically, let $\mathcal{I}$ denote all instances in the training and test set. For each instance $i \in \mathcal{I}$, we define a vector $\mathbf{x}_i = (\mathbf{x}_{\mathbf{GSR}i}, \mathbf{x}_{\mathbf{HR}i}, \mathbf{x}_{\mathbf{RR}i}, \mathbf{x}_{\mathbf{ST}i}) \in \Re^{120}$, where $\mathbf{x}_{\mathbf{GSR}i}, \mathbf{x}_{\mathbf{HR}i}, \mathbf{x}_{\mathbf{RR}i}, \mathbf{x}_{\mathbf{ST}i} \in \Re^{30}$, represent the raw vectors from the measurements of GSR, HR, RR, and ST, respectively, within the 30-s window for the same instance *i*. We denote the set of all these raw vectors by $\mathcal{X}_{\mathcal{I}} = \{\mathbf{x}_i | i \in \mathcal{I}\} \subset \Re^{120}$.

Certainly, the raw vectors generated in this way can readily be used in combination with state-of-the-art classification techniques with the goal of recognizing cognitive load (see also Section 6). However, given their high dimensionality, their time-series character, and the relatively small size of the training data set, i.e., low number of instances, it is often practical to transform the raw feature vectors into lower-dimensional feature vectors that carry the majority of important information from the raw data. This way, we can avoid the dimensionality problem, also known as the Curse of Dimensionality (Verleysen & François, 2005), and can develop highly discriminative and robust classifiers. To this end, we are interested in a transformation function $\Phi_{\mathcal{F}} : \mathcal{X} \rightarrow \Re^k, k \ll 120$, such that $\mathcal{F} = \{f_{GSR1}, f_{HR1}, f_{RR1}, f_{ST1}, \ldots, f_{GSRk}, f_{HRk}, f_{RRk}, f_{STk}\}$ is a family of feature functions of the form $f_{arg} : \Re^{30} \rightarrow \Re$, and

$$\begin{aligned}\Phi_{\mathcal{F}}(\mathbf{x}) \quad &= (f_{GSR1}(\mathbf{x}_{\mathbf{GSR}}), f_{HR1}(\mathbf{x}_{\mathbf{HR}}), f_{RR1}(\mathbf{x}_{\mathbf{RR}}), f_{ST1}(\mathbf{x}_{\mathbf{ST}}), \ldots, \\ &f_{GSRk}(\mathbf{x}_{\mathbf{GSR}}), f_{HRk}(\mathbf{x}_{\mathbf{HR}}), f_{RRk}(\mathbf{x}_{\mathbf{RR}}), f_{STk}(\mathbf{x}_{\mathbf{ST}}))\end{aligned}$$

There exist various possibilities to construct an adequate family of feature functions $\mathcal{F}$ for the described data set. Various approaches ranging from Wavelet and Fourier Transformations (Bloomfield, 2004; Chan & Fu, 1999; Chaovalit et al., 2011; Grinsted et al., 2004) to grammar-based evolutionary approaches (De Silva & Leong,) are able to deal with the time-series character of the data and generate low-dimensional feature vectors. For an overview, we refer the reader to (Fu, 2011). In fact, at least one of the competitors (Li & De Cock, 2020) in the CogLoad@UbiComp 2020 Competition builds on some of the mentioned feature generation strategies for the competing solution.

However, the relatively small size of the training data in the CogLoad@UbiComp 2020 Competition poses a serious challenge and does not allow these strategies to generate discriminatory features for robust predictions. Hence, after some analysis on feature generation strategies, we decided to use simple aggregation statistics as feature functions for $\mathcal{F}$. These include the *minimum*, the *maximum*, the *mean*, the *median*, the *standard deviation*, the *sum*, and the *skew* values from the 30-s sequence, for each of the 4 physiological signals (i.e., for GSR, HR, RR, and ST), respectively. The feature generation and model development pipeline for the solution presented in this article is depicted in Fig. 2.

Finally, we employed the CancelOut mechanism (Borisov et al., 2019) and the Shapley additive explanations (SHAP) framework (Lundberg & Lee, 2017) for the analysis of feature importance. The results of this analysis are shown in Fig. 5 and indicate that the vast majority of the proposed features (generated from the raw data) contain highly discriminative information for the prediction task at hand. As expected, the standard deviation of the physiological signals consistently appears among the most discriminative features.

5. A robust predictive method for cognitive load detection

In this section, we present our ensemble learning approach to robust and accurate cognitive load detection based on the CogLoad@UbiComp 2020 data set. The pipeline of feature generation and ensemble model development is summarized in Fig. 2.

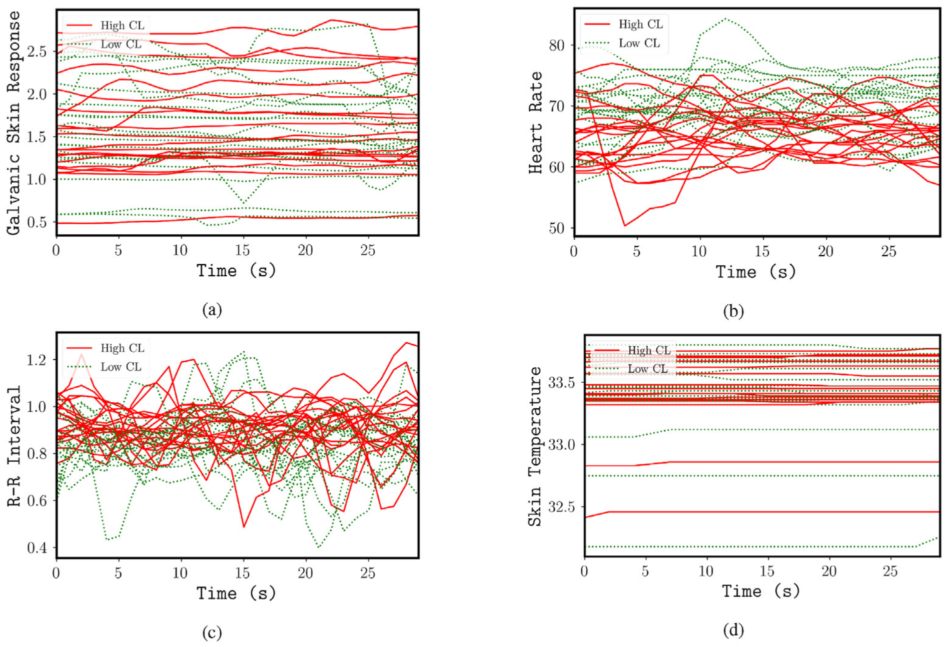


Fig. 1. Data set visualization for a single participant; the subfigures show galvanic skin responses (a), heart rates (b), R-R Intervals (c), and skin temperature (d). Red lines indicate a high cognitive load, and green lines represent a resting state, respectively. From the visual inspection, it can be seen that there are no clear distinctions between the two states. (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

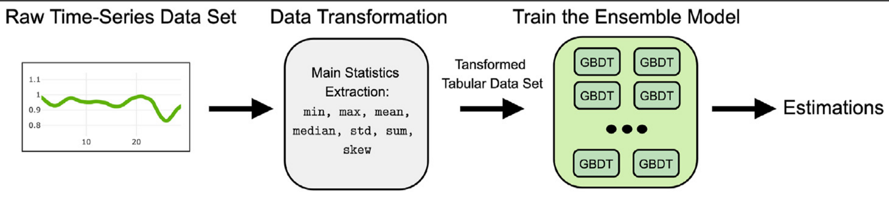

Raw Time-Series Data Set
Data Transformation
Train the Ensemble Model
Main Statistics Extraction:
min, max, mean, median, std, sum, skew
Tansformed Tabular Data Set
GBDT
Estimations

Fig. 2. Pipeline of feature generation and ensemble model development.

5.1. Base learners

After conducting an empirical evaluation of various machine algorithms as base learners for our ensemble approach, we selected the Gradient Boosting Decision Trees (GBDT) algorithm (Friedman, 2002). Our empirical findings on the excellent predictive performance of GBDT are also supported and complemented by previous results from numerous Data Science competitions and challenges. According to (Chen & Guestrin, 2016), in 2015, among the 29 winning solutions of Kaggle challenges (Kaggle.com), 17 solutions built on the GBDT algorithm.

5.2. Ensemble model

For the proposed ensemble solution, we used eight GBDT models. We built each of the eight models based on the LightGBM implementation (Ke et al., 2017). Prior studies have shown that the diversity of base learners clearly helps to reduce bias (which is very important for small data sets) and improve the overall performance of the ensemble algorithm (Kuncheva & Whitaker, 2003; Rokach, 2010). Thus, we trained the base learners by using different hyper-parameters. For the hyper-parameters selection, we utilized the random search and Bayesian optimization strategies (Bergstra et al., 2013). Since adding more models to the ensemble did not improve the predictive performance, the final meta-model consists of eight different GBDT models, where the final prediction is the mean value from all these models.

5.3. Validation

To provide robust estimations and exploit the training data as effectively as possible, we adopted an out-of-fold (OOF) cross-validation strategy. More specifically, from the folds that are used for validation during the cross-validation, we randomly generated hold-out samples, which served as unseen test examples. Based on these hold-out samples, we can estimate the predictive performance of each of the eight GBDT models on unseen data. For the cross-validation we employed a two iteration of stratified 5-fold cross-validation.

6. Experimental evaluation

In this section, we provide an overview of the experimental setup. More specifically, we describe the state-of-the-art predictive approaches that we have considered in the evaluation as baseline models and present the results of their predictive performance on the original feature vectors from $\mathcal{X}$, as well as on the transformed feature vectors using $\Phi_{\mathcal{F}}$ (as described in Section 4), in comparison to the approach proposed in this article.

6.1. State-of-the-art predictive algorithms as baseline approaches

In our experiments, we compare the proposed ensemble model with following predictive algorithms:

- Logistic Regression (LR) (Friedman et al., 2001). LR is a linear classifier with surprisingly strong predictive performance on many practical use cases. Moreover, the parameters of LR can also be fitted quite well on small-size training data.
- k-nearest neighbors (kNN) (Friedman et al., 2001). The kNN classifier often shows a nice predictive performance in practice and comes with strong theoretical guarantees on the possible classification error (more specifically, the Bayes error rate (Hastie et al., 2009)).
- Support-Vector-Machine (SVM) (Boser et al., 1992). The SVM classifier belongs to the most rigorously analysed and refined algorithms (e.g. (Hastie et al., 2009; Scholkopf & Smola, 2018),) and is often one of the best-performing predictive methods in practice.
- Random Forest (RF) (Breiman, 2001). The RF algorithm constructs an ensemble of decision trees that are sufficiently different from each other, allowing the RF to achieve a significantly higher predictive performance than individual decision trees (Hastie et al., 2009).
- Adaptive Boosting (AdaBoost) (Freund & Schapire, 1997). The AdaBoost algorithm is a highly versatile approach and, despite its simplicity, it works astonishingly well in practice, ranking it among the best performing ML methods.
- Gradient-Based Decision Trees (GBDT) (Friedman, 2002). The GBDT algorithm has proven to be one of the best performing predictive methods on heterogeneous tabular data (Chen & Guestrin, 2016). Its generalization and practical capabilities to handle missing values as well as variance and bias in the feature values make it one of the most valuable machine learning algorithms.
- Multilayer Perception (MLP) (Gardner & Dorling, 1998). Given the current popularity of neural networks, despite the relatively small data set, we felt that a corresponding approach should definitely be considered as a baseline. To this end, we decided to use a two-layer, fully-connected artificial neural network, the weights of which are optimized through stochastic gradient descent.

For all above algorithms, except GBDT, we used the scikit-learn library (Pedregosa et al., 2011). For GBDT, we selected the LightGBM implementation (Ke et al., 2017). For the hyper-parameters selection, we utilized the random search and Bayesian optimization strategies (Bergstra et al., 2013).

6.2. Evaluation measures

Following the instructions of the competition, two evaluation measures were used for the experiments:

- Accuracy, which is defined as $Accuracy = \frac{TP+TN}{TP+FN+TN+FP}$, where TP and TN are the true positive and true negative numbers, respectively. FP and FN represent the numbers of misclassifications for the negative and the positive class, respectively. Accuracy measures the fraction of correct predictions and works well for data sets in which the class frequencies are balanced, which is the case for the CogLoad@UbiComp 2020 data set. However, for imbalanced data sets, accuracy can be quite misleading.
- ROC-AUC (i.e., the area under the receiver operating characteristic curve), which quantifies the performance of a classification model over all classification score thresholds. The ROC curve plots two parameters: (1) the True Positive Rate, i.e., $tpr = \frac{TP}{TP+FN}$, and (2) the False Positive Rate, i.e., $fpr = \frac{FP}{FP+TN}$. Note that the tpr is a synonym for the recall of a predictive algorithm, whereas the fpr represents the rate of false alarms. The ROC curve plots the tpr vs. the fpr values at different

classification score thresholds. It can be shown that the area under the ROC curve is the ranking accuracy with respect to the classification score returned by a classifier. Ideally, instances that belong to the positive class should be assigned a higher score by the classifier and thus ranked higher than the instances that belong to the negative class. Hence, AUC of 1 means that all positive instances are ranked before the negative instances and the two classes are clearly separated by the classifier.

6.3. *Evaluation results*

In order to evaluate both the proposed ensemble method and the features generated by the proposed data transformation scheme, we conducted experiments using two data sets. The first data set consists of the original feature vectors, which are based on the raw (time-series). The second data set consists of the transformed feature vectors, i.e., using the $\Phi_{\mathcal{F}}$-transformation as described in Section 4.

All the evaluated models were developed and tuned on the same training data set based on a stratified 5-fold cross-validation. Note that because of the low number of participants, a stratification by participants leads to highly biased folds and does not allow the classifiers to generalize well to unseen data. Therefore, we only employed a class-based stratified sampling of the instances (i.e., feature vectors) for the 5 folds. The results of the evaluation are summarized in Table 1.

As shown in Table 1, the results of the comparison of the different approaches show that the proposed data transformation for the generation of new feature vectors consistently improves a model's predictive performance across all methods, regardless of the type of classification technique.

As was expected, the baseline approach based on the GBDT method shows an excellent predictive performance. It is in a tie with the SVM classifier on the original feature vectors and clearly outperforms all other baseline classifiers on the transformed data set (i.e. the transformed feature vectors using the $\Phi_{\mathcal{F}}$-transformation). The GBTD classifier is only outperformed by the proposed ensemble model. Consisting of several GBDT models, the proposed approach consistently shows the best predictive performance. It outperforms all the baseline classifiers by a considerable margin, both in terms of the Accuracy and the ROC-AUC measure. The ROC curves for the compared models are depicted in Fig. 3. They were generated on the validation sets of the transformed training data and clearly show the robust performance of the proposed ensemble method over all classification thresholds. Fig. 4 presents confusion matrices for all approaches which are used in the present study. When compared to other approaches submitted in the context of the CogLoad@UbiComp 2020 Challenge, our method returned the best predictions on unseen data, thus winning first place in the competition. Also, in comparison to the previous work on the cognitive load estimation using sensor information from a wrist-band device (e.g. (Gjoreski et al., 2018; Schaule et al., 2018)), our approach showed superior results.

Table 1
The performance comparison between various ML models on the original (raw) and the transformed data set. We evaluate the models using two iterations of stratified 5-fold cross-validation (2x5cv) with the following performance metrics: accuracy (higher is better) and ROC-AUC (higher is better) and report the mean and ± std results. The top results for each data set are marked in bold.

	Original Data Set		Transformed Data Set	
Model	Accuracy	ROC-AUC	Accuracy	ROC-AUC
LR	0.52 ± 0.030	0.54 ± 0.037	0.63 ± 0.025	0.65 ± 0.015
kNN	0.52 ± 0.028	0.54 ± 0.047	0.59 ± 0.023	0.61 ± 0.043
SVM	0.58 ± 0.025	0.60 ± 0.018	0.62 ± 0.040	0.65 ± 0.038
RF	0.56 ± 0.034	0.60 ± 0.036	0.64 ± 0.036	0.67 ± 0.043
AdaBoost	0.57 ± 0.039	0.57 ± 0.037	0.61 ± 0.030	0.64 ± 0.049
MLP	0.57 ± 0.038	0.60 ± 0.037	0.58 ± 0.030	0.60 ± 0.049
GBDT	0.58 ± 0.027	0.60 ± 0.036	0.65 ± 0.031	0.68 ± 0.034
Ensemble Model	**0.59 ± 0.035**	**0.61 ± 0.036**	**0.66 ± 0.035**	**0.69 ± 0.035**

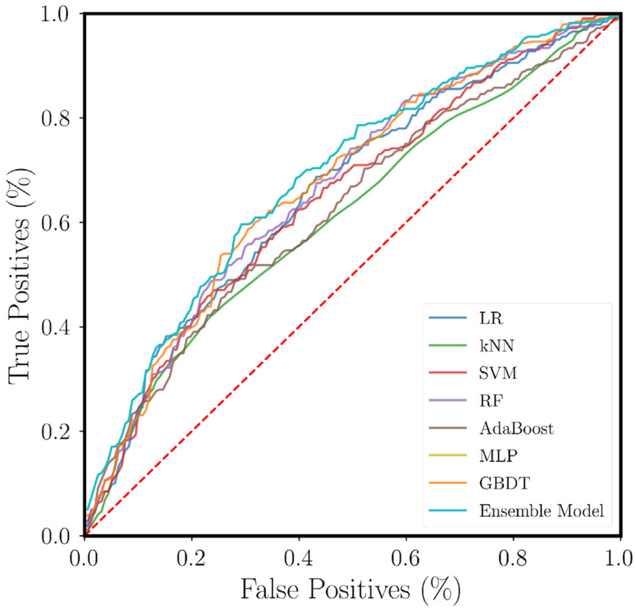


Fig. 3. ROC curves for all compared methods on the transformed data.

6.4. *Variable importance*

In Fig. 5, we present the feature importance analysis on the transformed data set. We employ the CancelOut neural layer proposed in (Borisov et al., 2019) and the Shapley additive explanations (SHAP) framework (Lundberg & Lee, 2017) to analyze the importance of the features for the prediction task at hand. Implemented as a neural layer, CancelOut was included as an additional layer to the MLP network. The SHAP analysis was conducted based on the GBDT model. The results depicted in Fig. 5 provide convincing evidence that the features produced by the standard deviation (std) function are among the most informative; there are three such features among the top-5 most important features. This is in line with our intuition and related work in the area of cognitive load detection which found that the variance of the measured physiological signals contains most of the information among the first three statistical moments of the (time-series) measures. The temperature sensor features, on the other hand, do not show a high relative importance. The RR- and HR-related features are the most informative according to the analysis, which is also quite intuitive, since the heart rate and its variability are known to be strongly correlated to cognitive load levels, e.g., (Gjoreski et al., 2018; Hughes et al., 2019; Wang & Guo, 2019).

Interestingly, the SHAP method assigns a higher importance to the skew-related features and lower importance to the mean-related features than the CancelOut method. Despite such differences, we can see that all the proposed transformations are valuable for the prediction of cognitive load from the available physiological signals.

7. Limitations and future work

Although our methods prove robust to subject-related bias and variability in the data, our results are based on a rather small data set derived from only 23 participants. Hence, further investigations on large data sets and higher variability are required. Of particular importance may be the investigation of inter-subject variability and whether or not and how the manifestation of cognitive load changes over time. Such an investigation requires, however, not only data collected in longitudinal studies, but also methods that are able to re-calibrate to subject-related

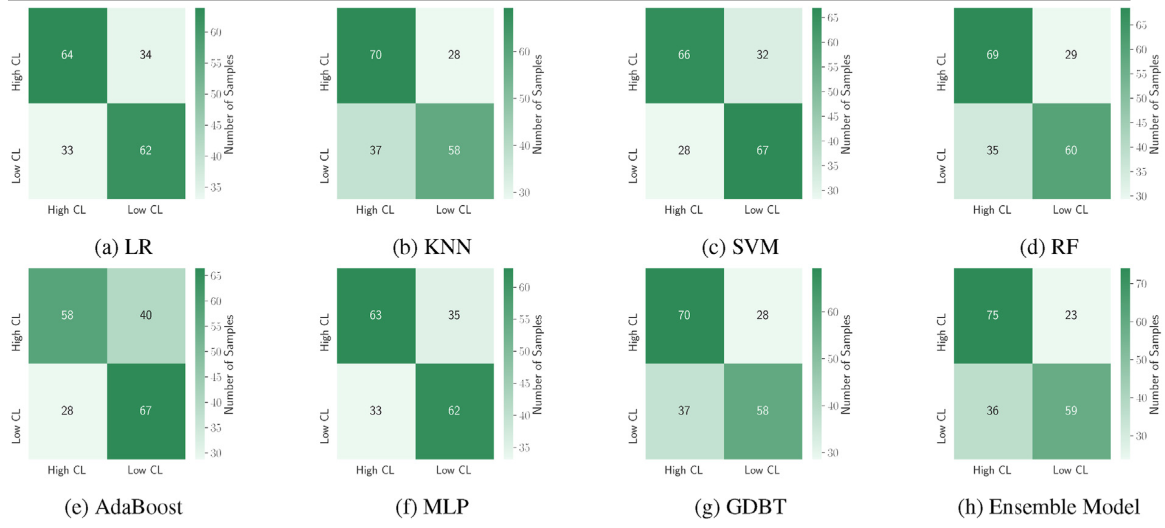


Fig. 4. Confusion matrices for baseline approaches and the proposed ensemble model on transformed test data set. As can be seen, our model shows superior performance among selected ML baseline in detection high cognitive load (High CL) by correctly identifying the state of the cognitive load; however, the SVM and Adaptive Boosting algorithms demonstrate the top performance in detecting the state where a subject has a regular or low cognitive load (Low CL). We use the same data splitting strategy as we did at the CogLoad@UbiComp 2020 competition, e. g in the test set, we have data for unseen subjects.

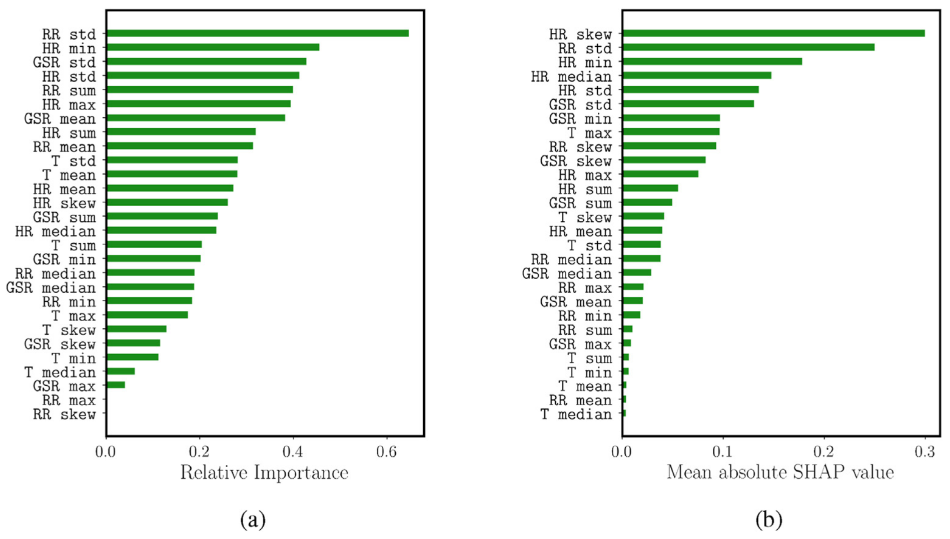


Fig. 5. Global variable importance analysis for the generated features using the CancelOut (a) and SHAP (b) algorithms.

characteristics.Regarding methodology, we also plan to investigate better ensemble strategies rather than simple mean aggregation. A better aggregation strategy for the ensemble step might improve the overall performance of our approach; one way is to rank models in the ensemble. A combination of the approaches might also help to estimate the cognitive load from a wearable device.

8. Conclusion

In this work, we presented an ensemble method for robust cognitive load detection based on physiological sensor data. With regard to its predictive power, our model outperformed the competing approaches, thus proving excellent robustness on unseen and small data. Furthermore, we showed that our proposed data transformation technique for the generation of new feature vectors from galvanic skin response, skin temperature, heart rate, and heart rate variability data notably improves the predictive power of machine learning techniques and might therefore be applicable to other areas of affective or cognition-aware computing. In contrast to the competing approaches, the effect of the input features on the model prediction is explainable, thus allowing a detailed analysis of cognitive load factors.

References

Abdelrahman, Y., Velloso, E., Dingler, T., Schmidt, A., & Vetere, F. (2017). Cognitive heat: Exploring the usage of thermal imaging to unobtrusively estimate cognitive load. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 1*, 1–20.

Amadieu, F., Van Gog, T., Paas, F., Tricot, A., & Mariné, C. (2009). Effects of prior knowledge and concept-map structure on disorientation, cognitive load, and learning. *Learning and Instruction, 19*, 376–386.

Appel, T., Scharinger, C., Gerjets, P., & Kasneci, E. (2018). Cross-subject workload classification using pupil-related measures. In *Proceedings of the 2018 ACM symposium on eye tracking research & applications* (pp. 1–8).

Appel, T., Sevcenko, N., Wortha, F., Tsarava, K., Moeller, K., Ninaus, M., Kasneci, E., & Gerjets, P. (2019). Predicting cognitive load in an emergency simulation based on

behavioral and physiological measures. In *2019 international conference on multimodal interaction* (pp. 154–163).
Babu, M. D., JeevithaShree, D., Prabhakar, G., Saluja, K. P. S., Pashilkar, A., & Biswas, P. (2019). Estimating pilots' cognitive load from ocular parameters through simulation and in-flight studies. *Journal of Eye Movement Research, 12*.
Beatty, J., Lucero-Wagoner, B., et al. (2000). The pupillary system. *Handbook of psychophysiology, 2*.
Benedetto, S., Pedrotti, M., & Bridgeman, B. (2011). Microsaccades and exploratory saccades in a naturalistic environment. *Journal of Eye Movement Research, 4*, 1–10.
Bergstra, J., Yamins, D., & Cox, D. D. (2013). Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms. In *Proceedings of the 12th Python in science conference, Citeseer* (p. 20).
van Berkel, N., Exler, A., Gjoreski, M., Kolenik, T., Okoshi, T., Pejovic, V., Visuri, A., Voit, A., & Ubittention. (2020). *5th international workshop on smart & ambient notification and attention management*.
Bloomfield, P. (2004). *Fourier analysis of time series: An introduction*. John Wiley & Sons.
Borisov, V., Haug, J., & Kasneci, G. (2019). Cancelout: A layer for feature selection in deep neural networks. In *International conference on artificial neural networks* (pp. 72–83). Springer.
Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory* (pp. 144–152).
Breiman, L. (2001). Random forests. *Machine Learning, 45*, 5–32.
Bristow, D., Frith, C., & Rees, G. (2005). Two distinct neural effects of blinking on human visual processing. *NeuroImage, 27*, 136–145.
Chan, K. P., & Fu, A. W. C. (1999). Efficient time series matching by wavelets. In *Proceedings 15th international conference on data engineering (cat. No. 99CB36337)* (pp. 126–133). IEEE.
Chaovalit, P., Gangopadhyay, A., Karabatis, G., & Chen, Z. (2011). Discrete wavelet transform-based time series analysis and mining. *ACM Computing Surveys, 43*, 1–37.
Chen, S., & Epps, J. (2014). Using task-induced pupil diameter and blink rate to infer cognitive load. *Human-Computer Interaction, 29*, 390–413.
Chen, S., Epps, J., Ruiz, N., & Chen, F. (2011). Eye activity as a measure of human mental effort in hci. In *Proceedings of the 16th international conference on Intelligent user interfaces* (pp. 315–318).
Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 785–794).
De Silva, A.M., Leong, P.H., . Grammar-based feature generation for time-series prediction. Springer.
Debie, E., Rojas, R. F., Fidock, J., Barlow, M., Kasmarik, K., Anavatti, S., & Abbass, H. A. (2021). Multimodal fusion for objective assessment of cognitive workload: A review. *IEEE Transactions on Cybernetics, 51*(3), 1542–1555. https://doi.org/10.1109/TCYB.2019.2939399
Di Nocera, F., Camilli, M., & Terenzi, M. (2007). A random glance at the flight deck: Pilots' scanning strategies and the real-time assessment of mental workload. *Journal of Cognitive Engineering and Decision Making, 1*, 271–285.
Duchowski, A., Krejtz, K., Krejtz, I., Biele, C., Niedzielska, A., Kiefer, P., Martin, R., & Giannopoulos, I. (2018). *The index of pupillary activity: Measuring cognitive load vis-à-vis task difficulty with pupil oscillation*. https://doi.org/10.1145/3173574.3173856
Faure, V., Lobjois, R., & Benguigui, N. (2016). The effects of driving environment complexity and dual tasking on drivers' mental workload and eye blink behavior. *Transportation Research Part F: Traffic Psychology and Behaviour, 40*, 78–90.
Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences, 55*, 119–139.
Fridman, L., Reimer, B., Mehler, B., & Freeman, W. T. (2018). Cognitive load estimation in the wild. In *Proceedings of the 2018 CHI conference on human factors in computing systems* (pp. 1–9). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/3173574.3174226.
Friedman, J. H. (2002). Stochastic gradient boosting. *Computational Statistics & Data Analysis, 38*, 367–378.
Friedman, N., Fekete, T., Gal, Y. K., & Shriki, O. (2019). Eeg-based prediction of cognitive load in intelligence tests. *Frontiers in Human Neuroscience, 13*, 191.
Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning* (Vol. 1). New York: Springer series in statistics.
Fu, T.c. (2011). A review on time series data mining. *Engineering Applications of Artificial Intelligence, 24*, 164–181.
Fukuda, K., Stern, J. A., Brown, T. B., & Russo, M. B. (2005). Cognition, blinks, eye-movements, and pupillary movements during performance of a running memory task. *Aviation Space & Environmental Medicine, 76*, C75–C85.
Gao, X., Yan, H., & Sun, H.j. (2015). Modulation of microsaccade rate by task difficulty revealed through between-and within-trial comparisons. *Journal of Vision, 15*, 3–3.
Gardner, M. W., & Dorling, S. (1998). Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmospheric Environment, 32*, 2627–2636.
Gjoreski, M., Kolenik, T., Knez, T., Luštrek, M., Gams, M., Gjoreski, H., & Pejović, V. (2020). Datasets for cognitive load inference using wearable sensors and psychological traits. *Applied Sciences, 10*, 3843.
Gjoreski, M., Luštrek, M., & Pejović, V. (2018). My watch says i'm busy: Inferring cognitive load with low-cost wearables. In *Proceedings of the 2018 ACM international joint conference and 2018 international symposium on pervasive and ubiquitous computing and wearable computers* (pp. 1234–1240).
Grinsted, A., Moore, J. C., & Jevrejeva, S. (2004). *Application of the cross wavelet transform and wavelet coherence to geophysical time series*.
Grubov, V., Badarin, A., & Maksimenko, V. (2020). Analysis of information perception and processing during long-term and intense cognitive load using combined eeg and nirs. In *2020 international conference nonlinearity, information and robotics (NIR)* (pp. 1–2). IEEE.
Hart, S. G., & Staveland, L. E. (1988). Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Advances in psychology* (Vol. 52, pp. 139–183). Elsevier.
Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction*. Springer Science & Business Media.
He, J., & McCarley, J. S. (2010). Executive working memory load does not compromise perceptual processing during visual search: Evidence from additive factors analysis. *Attention, Perception, & Psychophysics, 72*, 308–316.
Hogervorst, M. A., Brouwer, A. M., & van Erp, J. B. F. (2014). Combining and comparing eeg, peripheral physiology and eye-related measures for the assessment of mental workload. *Frontiers in Neuroscience, 8*, 322. https://doi.org/10.3389/fnins.2014.00322, 25352774[pmid] http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4196537/.
Huang, W., Li, J., & Alem, L. (2018). Towards preventative healthcare: A review of wearable and mobile applications. *Data, Informatics and Technology: An Inspiration for Improved Healthcare*, 11–14.
Hughes, A. M., Hancock, G. M., Marlow, S. L., Stowers, K., & Salas, E. (2019). Cardiac measures of cognitive workload: A meta-analysis. *Human Factors, 61*, 393–414.
Inamdar, S., & Pomplun, M. (2003). Comparative search reveals the tradeoff between eye movements and working memory use in visual tasks. In *Proceedings of the annual meeting of the cognitive science society*.
Kagglecom. Kaggle competitions. URL https://www.kaggle.com/competitions.
Kahneman, D. (1973). *Attention and effort* (Vol. 1063). Citeseer.
Kasneci, E., Kübler, T., Broelemann, K., & Kasneci, G. (2017). Aggregating physiological and eye tracking signals to predict perception in the absence of ground truth. *Computers in Human Behavior, 68*, 450–455.
Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in neural information processing systems* (pp. 3146–3154).
Keshmiri, S., Sumioka, H., Yamazaki, R., & Ishiguro, H. (2017). A non-parametric approach to the overall estimate of cognitive load using nirs time series. *Frontiers in Human Neuroscience, 11*, 15.
Kosch, T., Hassib, M., Woźniak, P. W., Buschek, D., & Alt, F. (2018). Your eyes tell: Leveraging smooth pursuit for assessing cognitive workload. In *Proceedings of the 2018 CHI conference on human factors in computing systems* (pp. 1–13).
Kramer, A. E. (1990). *Physiological metrics of mental workload: A review of recent progress*.
Kramer, A. F. (1991). Physiological metrics of mental workload: A review of recent progress. *Multiple-task Performance*, 279–328.
Krejtz, K., Duchowski, A. T., Niedzielska, A., Biele, C., & Krejtz, I. (2018). Eye tracking cognitive load using pupil diameter and microsaccades with fixed gaze. *PloS One, 13*, Article e0203629.
Kübler, T. C., Kasneci, E., Rosenstiel, W., Schiefer, U., Nagel, K., & Papageorgiou, E. (2014). Stress-indicators and exploratory gaze for the analysis of hazard perception in patients with visual field loss. *Transportation Research Part F: Traffic Psychology and Behaviour, 24*, 231–243.
Kuncheva, L. I., & Whitaker, C. J. (2003). Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning, 51*, 181–207.
Kun, A. L., Palinko, O., Medenica, Z., & Heeman, P. A. (2013). *On the feasibility of using pupil diameter to estimate cognitive load changes for in-vehicle spoken dialogues*. INTERSPEECH.
Li, X., & De Cock, M. (2020). Cognitive load detection from wrist-band sensors. In *Adjunct proceedings of the 2020 ACM international joint conference on pervasive and ubiquitous computing and proceedings of the 2020 ACM international symposium on wearable computers* (pp. 456–461).
Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 30, pp. 4765–4774). Curran Associates, Inc.. URL http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf.
Mäki-Marttunen, V., Hagen, T., & Espeseth, T. (2019). Task context load induces reactive cognitive control: An fmri study on cortical and brain stem activity. *Cognitive, Affective, & Behavioral Neuroscience, 19*, 945–965.
Marshall, S. P. (2000). Method and apparatus for eye tracking and monitoring Pipil dilation to evaluate cognitive activity. URL https://patentimages.storage.googleapis.com/pdfs/9171d27ab488a900c7db/US6090051.pdf.
Marshall, S. P. (2007). Identifying cognitive state from eye metrics. *Aviation Space & Environmental Medicine, 78*, B165–B175.
McDuff, D., Gontarek, S., & Picard, R. (2014). Remote measurement of cognitive stress via heart rate variability. In *2014 36th annual international conference of the IEEE engineering in medicine and biology society, IEEE* (pp. 2957–2960).
Mehler, B., Reimer, B., & Wang, Y. (2011). *A comparison of heart rate and heart rate variability indices in distinguishing single-task driving and driving under secondary cognitive workload*.
Mills, C., Fridman, I., Soussou, W., Waghray, D., Olney, A. M., & D'Mello, S. K. (2017). Put your thinking cap on: Detecting cognitive load using eeg during learning. In *Proceedings of the seventh international learning analytics & knowledge conference* (pp. 80–89).
Nourbakhsh, N., Chen, F., Wang, Y., & Calvo, R. A. (2017). Detecting users' cognitive load by galvanic skin response with affective interference. *ACM Transactions on Interactive Intelligent Systems (TiiS), 7*, 1–20.

Nourbakhsh, N., Wang, Y., Chen, F., & Calvo, R. A. (2012). Using galvanic skin response for cognitive load measurement in arithmetic and reading tasks. In *Proceedings of the 24th Australian computer-human interaction conference* (pp. 420–423).

Palinko, O., Kun, A. L., Shyrokov, A., & Heeman, P. (2010). Estimating cognitive load using remote eye tracking in a driving simulator. In *Proceedings of the 2010 symposium on eye-tracking research & applications, ACM* (pp. 141–144).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *The Journal of Machine Learning Research, 12*, 2825–2830.

Pfleging, B., Fekety, D. K., Schmidt, A., & Kun, A. L. (2016). A model relating pupil diameter to mental workload and lighting conditions. In *Proceedings of the 2016 CHI conference on human factors in computing systems* (pp. 5776–5788).

Prabhakar, G., Mukhopadhyay, A., Murthy, L., Modiksha, M., Sachin, D., & Biswas, P. (2020). Cognitive load estimation using ocular parameters in automotive. *Transport Engineer, 2*, 100008.

Rafiqi, S., Wangwiwattana, C., Fernandez, E., Nair, S., & Larson, E. (2015). Work-in-progress, pupilware-m: Cognitive load estimation using unmodified smartphone cameras. In *2015 IEEE 12th international conference on mobile ad hoc and sensor systems* (pp. 645–650). https://doi.org/10.1109/MASS.2015.31

Rayner, K. (1998). Eye movements in reading and information processing: 20 years of research. *Psychological Bulletin, 124*, 372.

Rokach, L. (2010). Ensemble-based classifiers. *Artificial Intelligence Review, 33*, 1–39.

Salfinger, A. (2020). Deep learning for cognitive load monitoring: A comparative evaluation. In *Adjunct proceedings of the 2020 ACM international joint conference on pervasive and ubiquitous computing and proceedings of the 2020 ACM international symposium on wearable computers* (pp. 462–467).

Schaule, F., Johanssen, J. O., Bruegge, B., & Loftness, V. (2018). Employing consumer wearables to detect office workers' cognitive load for interruption management. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 2*, 1–20.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks, 61*, 85–117.

Schmiedek, F., Lövdén, M., & Lindenberger, U. (2014). A task is a task is a task: Putting complex span, n-back, and other working memory indicators in psychometric context. *Frontiers in Psychology, 5*, 1475.

Scholkopf, B., & Smola, A. J. (2018). *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. Adaptive Computation and Machine Learning series.

Setz, C., Arnrich, B., Schumm, J., La Marca, R., Tröster, G., & Ehlert, U. (2010). Discriminating stress from cognitive load using a wearable eda device. *IEEE Transactions on Information Technology in Biomedicine, 14*, 410–417. https://doi.org/10.1109/TITB.2009.2036164

Sharma, K., Niforatos, E., Giannakos, M., & Kostakos, V. (2020). Assessing cognitive performance using physiological and facial features: Generalizing across contexts. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 4*, 1–41.

Shi, Y., Ruiz, N., Taib, R., Choi, E., & Chen, F. (2007). Galvanic skin response (gsr) as an index of cognitive load. In *CHI'07 extended abstracts on Human factors in computing systems* (pp. 2651–2656).

Siegenthaler, E., Costela, F. M., McCamy, M. B., Di Stasi, L. L., Otero-Millan, J., Sonderegger, A., Groner, R., Macknik, S., & Martinez-Conde, S. (2014). Task difficulty in mental arithmetic affects microsaccadic rates and magnitudes. *European Journal of Neuroscience, 39*, 287–294.

Stubbs, J. L., Corrow, S. L., Kiang, B., Panenka, W. J., & Barton, J. J. (2018). The effects of enhanced attention and working memory on smooth pursuit eye movement. *Experimental Brain Research, 236*, 485–495.

Sweller, J. (2011). Cognitive load theory. In *Psychology of learning and motivation* (Vol. 55, pp. 37–76). Elsevier.

international joint conference on pervasive, T.A., ubiquitous computing. (2020). Ubicomp2020. URL https://ubicomp.hosting.acm.org/ubicomp2020/.

Tervonen, J., Pettersson, K., & Mäntyjärvi, J. (2021). Ultra-short window length and feature importance analysis for cognitive load detection from wearable sensors. *Electronics, 10*, 613.

Van Gog, T., Paas, F., & Van Merriënboer, J. J. (2005). Uncovering expertise-related differences in troubleshooting performance: Combining eye movement and concurrent verbal protocol data. *Applied Cognitive Psychology, 19*, 205–221.

Van Orden, K. F., Limbert, W., Makeig, S., & Jung, T. P. (2001). Eye activity correlates of workload during a visuospatial memory task. *Human Factors, 43*, 111–121.

Verleysen, M., & François, D. (2005). The curse of dimensionality in data mining and time series prediction. In *International work-conference on artificial neural networks* (pp. 758–770). Springer.

Wang, C., & Guo, J. (2019). A data-driven framework for learners' cognitive load detection using ecg-ppg physiological feature fusion and xgboost classification. *Procedia computer science, 147*, 338–348.

Wang, Q., Yang, S., Liu, M., Cao, Z., & Ma, Q. (2014). An eye-tracking study of website complexity from cognitive load perspective. *Decision Support Systems, 62*, 1–10. https://doi.org/10.1016/j.dss.2014.02.007. URL http://www.sciencedirect.com/science/article/pii/S0167923614000402.

Wilson, G., Fullenkamp, P., & Davis, I. (1994). Evoked potential, cardiac, blink, and respiration measures of pilot workload in air-to-ground missions. *Aviation Space & Environmental Medicine, 65 2*, 100–105.

A.3.2 BoxShrink: From Bounding Boxes to Segmentation Masks

Publication: Published in the Medical Image Learning with Limited and Noisy Data (MILLanD) workshop at the Medical Image Computing and Computer Assisted Intervention (MICCAI) conference, 2022.

Contribution: Michael Gröger and I came up with an idea of using superpixes and conditional random fields for weakly-supersized object detection tasks. Michael Gröger developed all major parts of the explainability framework and the implementations of the relational local explanation framework. Gjergji Kasneci contributed to the paper by challenging and improving ideas, formalizations, and the analysis of results. All co-authors helped revise the final manuscript.

BoxShrink: From Bounding Boxes to Segmentation Masks

Michael Gröger(✉), Vadim Borisov, and Gjergji Kasneci

University of Tübingen, Tübingen, Germany
michael.groeger@posteo.net

Abstract. One of the core challenges facing the medical image computing community is fast and efficient data sample labeling. Obtaining fine-grained labels for segmentation is particularly demanding since it is expensive, time-consuming, and requires sophisticated tools. On the contrary, applying bounding boxes is fast and takes significantly less time than fine-grained labeling, but does not produce detailed results. In response, we propose a novel framework for weakly-supervised tasks with the rapid and robust transformation of bounding boxes into segmentation masks without training any machine learning model, coined BoxShrink. The proposed framework comes in two variants – *rapid*-BoxShrink for fast label transformations, and *robust*-BoxShrink for more precise label transformations. An average of four percent improvement in IoU is found across several models when being trained using BoxShrink in a weakly-supervised setting, compared to using only bounding box annotations as inputs on a colonoscopy image data set. We open-sourced the code for the proposed framework and published it online.

Keywords: Weakly-supervised learning · Segmentation · Colonoscopy · Deep neural networks

1 Introduction

Convolutional neural networks (CNNs) have achieved remarkable results across image classification tasks of increasing complexity, from pure image classification to full panoptic segmentation, and have become, as a consequence, the standard method for these tasks in computer vision [19]. However, there are also certain drawbacks associated with these methods. One of them is that in order to achieve satisfactory results, a data set of an appropriate size and high-quality labels are needed [21]. The costs and time associated with labeling increase with the complexity of the task, with image classification being the cheapest and image segmentation being the most expensive one. All of these challenges especially

M. Gröger and V. Borisov—equal contribution.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-16760-7_7.

G. Zamzmi et al. (Eds.): MILLanD 2022, LNCS 13559, pp. 65–75, 2022.
https://doi.org/10.1007/978-3-031-16760-7_7

apply to medical artificial intelligence (MAI) applications since they depend on the input and feedback by expensive domain experts [22].

In this work, we present a novel approach for fast segmentation label preprossessing, *which is decoupled from any particular artificial neural network architecture.* The proposed algorithmic framework can serve as a first approach for practitioners to transform a data set with only bounding box annotations into a prelabeled (i.e., semantically segmented) version of the data set. Our framework consists of independent components such as superpixels [23], fully-connected conditional random fields [14] and embeddings. This makes it easy to add our framework to an existing machine learning pipeline.

To evaluate the proposed framework, we select an endoscopic colonoscopy data set [4]. Multiple experiments show that our framework helps to considerably reduce the gap between the segmentation performance and efficiency of a neural network that is trained only on bounding boxes and one trained on fully segmented segmentation masks.

The main contributions of this work are:

– We propose the BoxShrink framework consisting of two methods. One for a time-efficient and one for a more robust transformation of bounding-boxes into segmentation masks. In both methods there is no need to train a model.
– We publish our bounding-box labels for the CVC-Clinic data set for future research in the area of weakly-supervised learning.
– We open-source our code and publish it online.[1]

2 Related Work

In this Section, we further define weakly-supervised learning and separate it from other approaches such as semi-supervised learning. Also, we localize our work among those which use similar components.

To reduce the need for resources such as time and money, various learning methodologies were introduced such as semi-supervised and weakly-supervised learning [30]. Semi-supervised learning leverages labeled data, e.g. for segmentation tasks correctly and fully segmented images and the availability of a larger amount of unlabeled data [16]. Weakly-supervised learning on the other hand, exploits noisy labels as a weak supervisory signal to generate segmentation masks. These labels can be provided in different forms such as points [3], or image-level labels [27], being the more simpler ones, or more complex ones such as scribbles [15,24], or bounding boxes [6,11]. A similar work [29] to ours also utilizes superpixel embeddings and CRFs, but their method requires an additional construction of a graph of superpixels and a custom deep neural network architecture. Our method, on the other hand, is easier to integrate into existing pipelines. Also, in contrast to many other weakly-supervised approaches [10,28], we do not apply CRFs as a postprocessing step on the output of the model but as a preprocessing step on the input, hence, we leave the downstream model untouched. Furthermore, the proposed framework does not require special hardware such as GPU or TPU for the label preprocessing step.

[1] https://github.com/michaelgroeger/boxshrink.

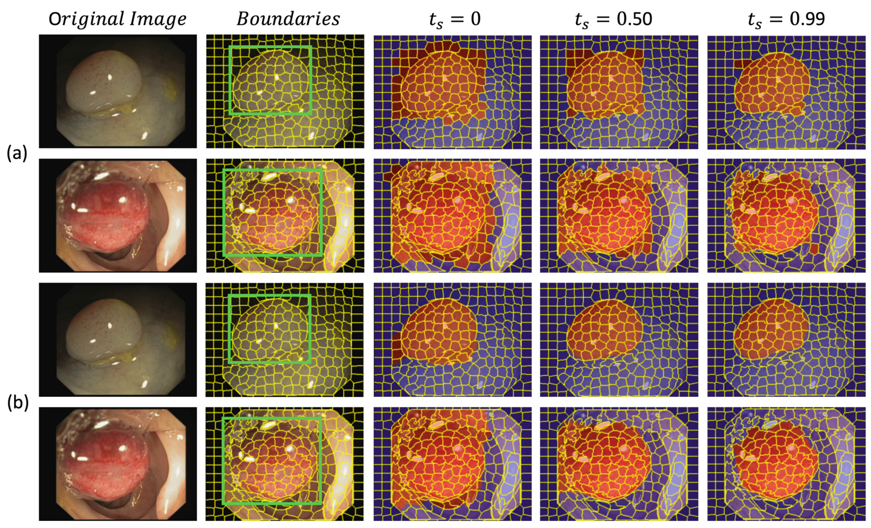


Fig. 1. The impact of varying the threshold t_s, i.e., a hyperparameter of the BoxShrink framework for tuning the final segmentation quality, where (a) shows two data samples from the data set after the superpixel assignment step (Sect. 3.2), and (b) demonstrates pseudo-masks after the FCRF postprocessing. As seen from this experiment, having a higher threshold might generate better masks but increases the risk of losing correct foreground pixels.

3 Boxshrink Framework

This section presents our proposed BoxShrink framework. First, we define its main components: superpixel segmentation, fully-connected conditional random fields, and the embedding step. We then explain two different settings of the framework, both having the same goal: to reduce the number of background pixels labeled as foreground contained in the bounding box mask.

3.1 Main Components

Superpixels aim to group pixels into bigger patches based on their color similarity or other characteristics [23]. In our implementation, we utilize the SLIC algorithm proposed by [1] which is a k-means-based algorithm grouping pixels based on their proximity in a 5D space. A crucial hyperparameter of SLIC is the number of segments to be generated which is a upper bound for the algorithm on how many superpixels should be returned for the given image. The relationship between the output of SLIC and the maximum number of segments can be seen in the supplementary material.

Fully-connected-CRFs are an advanced version of conditional random fields (CRFs) which represent pixels as a graph structure. CRFs take into account a unary potential of each pixel and the dependency structure between that pixel and its neighboring ones using pairwise potentials [25]. Fully-connected-CRFs (FCRFs) address some of the limitations of classic CRFs, such as the lack of capturing long-range dependencies by connecting all pixel pairs. Equation 1 shows the main building block of FCRFs which is the Gibbs-Energy function [13].

$$E(x) = \sum_{i=1}^{N} \psi_u(x_i) + \sum_{i<j}^{N} \psi_p(x_i, x_j), \tag{1}$$

where the first term $\psi_u(x_i)$ measures the unary potential, that is, the cost if the assigned label disagrees with that of the initial classifier, the second term $\psi_p(x_i, x_j)$ measures the pairwise potential, which is the cost if two similar pixels disagree on their label x. The input is over all pixels N. We use FCRFs to smooth the output pseudo-mask.

Superpixel Embeddings are a key component of the *robust*-BoxShrink variant. The embedding function M produces a numerical representation of every superpixel $k_i \in \boldsymbol{K}$ by returning an embedding vector. Formally, this operation can be depicted $M : \mathbb{R}^m \rightarrow \mathbb{R}^n$. Practically, this can be done by feeding each superpixel k_i separately into a CNN model, such as a Resnet-50 [9] pretrained on ImageNet [7]. By doing so, we obtain a 2048-dimensional vector representation for every superpixel. It allows us to get an aggregated representation of the foreground and background, by computing the mean embedding of all foreground and background superpixels in the training data set. These mean vectors are then used to assign superpixels either to the foreground or background class based on their cosine similarity.

3.2 *rapid*-BoxShrink

We first split each image into superpixels using the SLIC algorithm for the *rapid*-BoxShrink strategy. We overlap the superpixels with the provided bounding box mask and build a new mask based on those superpixels, which overlap the bounding box mask to a certain threshold. This approach is based on the assumption that the object of interest is always fully contained in the bounding box. The results depend on the number of segments generated which can be seen in the supplementary materials and the chosen threshold shown in Fig. 1. To this end, as shown in the supplementary material in Algorithm 1, to make the final pseudo-mask more smooth, we run a FCRF as described in Sect. 3.1 on the thresholded superpixel mask.

3.3 *robust*-BoxShrink

Leveraging the availability of superpixels, we also explore the use of embeddings to shrink the number of background pixels in the pseudo-mask. We segmented each image in the training data set into superpixels and then assigned them either to the foreground or background group by applying the thresholding approach as we have done it in the *rapid*-BoxShrink variant (Sect. 3.2). To

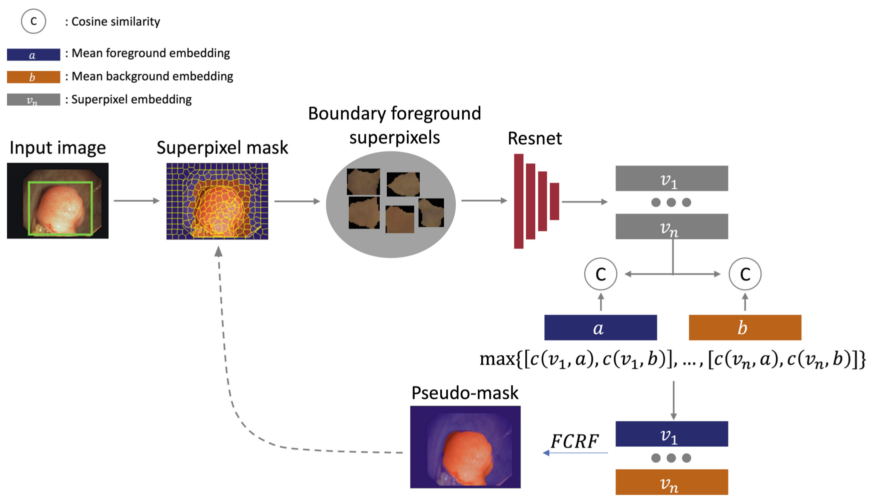

C : Cosine similarity
a : Mean foreground embedding
b : Mean background embedding
v_n : Superpixel embedding
Input image
Superpixel mask
Boundary foreground superpixels
Resnet
v_1
v_n
C
C
a
b
$\max\{[c(v_1,a),c(v_1,b)],\ldots,[c(v_n,a),c(v_n,b)]\}$
Pseudo-mask
FCRF
v_1
v_n

Fig. 2. Overview of the *robust*-BoxShrink method assuming the mean embedding vectors are given. First, we generate a superpixel mask based on the *rapid*-BoxShrink approach but without utilizing the FCRF. Then, we extract each foreground superpixel on the boundary between foreground and background. Feeding each superpixel into a pretrained ResNet model yields one 2048-dimensional embedding vector per superpixel. Next, we calculate the cosine similarity score of each embedding and the mean background and foreground embedding. Based on the highest score we either keep the superpixel as foreground or assign it to the background class. Finally, we apply a FCRF on the resulting superpixel mask. The dashed line indicates that this approach can be run iteratively.

generate the pseudo-masks, we start with the bounding box mask and segment the image using again the thresholding technique. This yields $\mathcal{F}$ superpixels for the foreground and $\mathcal{B}$ superpixels for the background. Then we go along the boundary foreground superpixels $\mathcal{F}_o$ and assign them either to the background or foreground class, depending on their cosine similarity score to the mean background and foreground embedding. The whole process can be seen in Fig. 2. The Algorithm 2, which can be found in the supplementary materials, summarizes the main steps of the *robust*-BoxShrink method.

4 Experiments

This Section presents qualitative and quantitative experiments for both versions of the BoxShrink framework.

Data Set. For all our experiments we utilize the endoscopic colonoscopy frames for polyp detection data set (CVC-Clinic DB) [4], it consists of 612 endoscopy images, each having a size of 288 × 384 × 3. The data set comes along with binary ground truth segmentation masks, which we utilize for the evaluation of

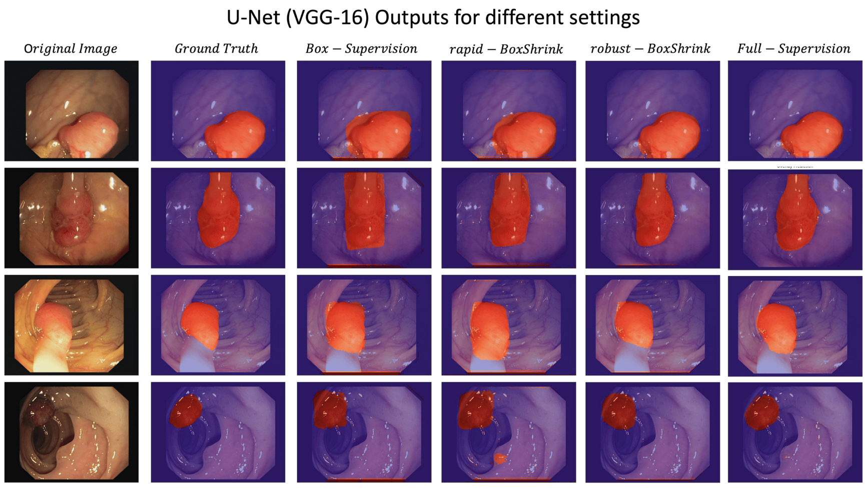


Fig. 3. Qualitative model prediction masks on four random samples from the CVC-Clinic test set. The setting on which the model was trained on is indicated on top.

our weakly-supervised framework and to infer the bounding boxes. This data set was featured in multiple studies [2,8].

4.1 Qualitative and Quantitative Experiments

For our experiments, we utilize two popular deep learning architectures for segmentation tasks - U-Net [20] and DeepLabV3+ [5].

Settings. We have four settings, using: (1) Bounding boxes as labels which serves as our lower baseline, (2) labels generated with the *rapid*-BoxShrink label transformation strategy, (3) labels generated with the *robust*-BoxShrink label transformation strategy, and (4) a fully-supervised upper baseline with segmentation masks as labels.

Quality Measure. We use the Intersection over Union (IoU) score as an evaluation measure. The IoU, also called Jaccard similarity J between two sets $\boldsymbol{A}$ and $\boldsymbol{B}$, is a commonly used measure of how well the prediction aligns with the ground truth in image segmentation [18]. As the equation below shows, the IoU is computed by dividing the intersection of two masks by their union.

$$J(\boldsymbol{A}, \boldsymbol{B}) = \frac{|\boldsymbol{A} \cap \boldsymbol{B}|}{|\boldsymbol{A} \cup \boldsymbol{B}|}. \tag{2}$$

Results. We present the quantitative results in Table 1. In line with other publications, we also share situations where our presented Framework fails. Figure 5, which can be found in the supplementary materials shows some examples.

Figure 3 shows some good prediction masks from the test set made by models trained on the aforementioned four different settings.

4.2 Reproducibility Details

We split the CVC-Clinic DB data set into 80% training data, 10% validation data and 10% test data. For splitting, we use the implementation from sklearn [17] with a random state of 1. To generate the superpixel masks, we set the maximum number of segments s to 200, a threshold t_s of 0.6 for all training images and use the implementation from skimage [26]. To get the embeddings, we use a maximum number s of 250 segments and a threshold t_s of 0.1 to not loose too much of the foreground. To smooth the superpixel masks we use the FCRF implementation provided by the pydensecrf package.[2] Note that we do not train the FCRF (similar to [10]) and set the FCRF hyperparmeters of the x/y-standard deviation for the pairwise Gaussian to 5 and for the pairwise bilateral to 25. We set the rgb-standard deviation to 10. To determine the best performing model, we use the intersection over union (IoU) during training on the validation set. After the training, the best performing model is kept and evaluated once on the test set. Both, the test and validation set consist of ground truth masks. We generate all models using the segmentation-models PyTorch library.[3]

For our experiments we select ResNet-18, ResNet-50, and VGG-16 backbones pretrained on the ImageNet data set paired with U-Net and DeepLabV3+ as a decoder. We use the Sigmoid function as an activation function and the Adam [12] optimizer with a learning rate of 0.0001. As the loss function we utilize the Cross-Entropy Loss. During training, we apply step-wise learning rate scheduling where we decay the learning rate by 0.5 each 5 epochs. We train the ResNet-18 & VGG-16 architecture for 25 epochs and the ResNet-50 architecture for 15 epochs. The training is being done on a 16 GB Nvidia Tesla P-100. We use a batch size of 64 when using the ResNet-18, 32 for the VGG-16 architecture and 16 when using ResNet-50. For both methods, *rapid*-BoxShrink and *robust*-BoxShrink, we return the initial bounding box mask if the total mask occupancy, that is the ratio of the bounding box and the total image is less than 0.1 or the IoU between the pseudo mask and the bounding box mask is less than 0.1.

5 Discussion

In this Section, we further discuss the application and future work of the proposed weakly-supervised framework.

The choice between *rapid*-BoxShrink and *robust*-BoxShrink depends on multiple factors - the time budget and expected label transformation quality. In our experiments, we observe that *rapid*-BoxShrink takes on average only 0.5 seconds to transform the labels for a singe data sample, where *robust*-BoxShrink

[2] https://github.com/lucasb-eyer/pydensecrf.
[3] https://github.com/qubvel/segmentation_models.pytorch.

Table 1. Experimental results on the CVC-Clinic data set. All models are evaluated on the *ground truth segmentation mask* in the validation and test set. The label format indicates the initial input label on which the model was either trained or our proposed frameworks were applied to. The results are averages of six runs; we also report the corresponding standard deviation for each setting. This is being done to deliver a more consistent picture because of the random initialization of the decoder part and the stochasticity of the optimizer. The best performing results for our proposed methods are marked in bold. Higher IoU is better.

Segmentation model	Label format	Backbone	Validation (IoU)	Test (IoU)
U-Net	Bounding Boxes	VGG-16	0.749 ± 0.023	0.772 ± 0.030
U-Net (*rapid*-BoxShrink)	Bounding Boxes	VGG-16	0.769 ± 0.026	0.807 ± 0.028
U-Net (*robust*-BoxShrink)	Bounding Boxes	VGG-16	$\mathbf{0.775} \pm 0.013$	$\mathbf{0.824} \pm 0.010$
U-Net	Segment. Masks	VGG-16	0.796 ± 0.025	0.829 ± 0.025
U-Net	Bounding Boxes	ResNet-18	0.691 ± 0.051	0.729 ± 0.060
U-Net (*rapid*-BoxShrink)	Bounding Boxes	ResNet-18	0.730 ± 0.021	0.781 ± 0.024
U-Net (*robust*-BoxShrink)	Bounding Boxes	ResNet-18	$\mathbf{0.755} \pm 0.021$	$\mathbf{0.808} \pm 0.021$
U-Net	Segment. Masks	ResNet-18	0.800 ± 0.032	0.859 ± 0.044
U-Net	Bounding Boxes	ResNet-50	0.785 ± 0.010	0.810 ± 0.010
U-Net (*rapid*-BoxShrink)	Bounding Boxes	ResNet-50	0.807 ± 0.018	0.851 ± 0.019
U-Net (*robust*-BoxShrink)	Bounding Boxes	ResNet-50	$\mathbf{0.813} \pm 0.015$	$\mathbf{0.852} \pm 0.012$
U-Net	Segment. Masks	ResNet-50	0.889 ± 0.012	0.920 ± 0.016
DeepLabV3+	Bounding Boxes	VGG-16	0.746 ± 0.033	0.766 ± 0.034
DeepLabV3+ (*rapid*-BoxShrink)	Bounding Boxes	VGG-16	$\mathbf{0.779} \pm 0.023$	$\mathbf{0.817} \pm 0.0201$
DeepLabV3+ (*robust*-BoxShrink)	Bounding Boxes	VGG-16	0.767 ± 0.0187	0.809 ± 0.024
DeepLabV3+	Segment. Masks	VGG-16	0.832 ± 0.049	0.858 ± 0.051
DeepLabV3+	Bounding Boxes	ResNet-18	0.723 ± 0.025	0.758 ± 0.021
DeepLabV3+ (*rapid*-BoxShrink)	Bounding Boxes	ResNet-18	0.743 ± 0.021	0.787 ± 0.026
DeepLabV3+ (*robust*-BoxShrink)	Bounding Boxes	ResNet-18	$\mathbf{0.759} \pm 0.005$	$\mathbf{0.806} \pm 0.002$
DeepLabV3+	Segment. Masks	ResNet-18	0.808 ± 0.010	0.844 ± 0.012

needs on average 3 seconds to complete the label transformation, the processing time can be further optimized in future versions. However, from our extensive experiments (Sect. 4.1), we can conclude that *robust*-BoxShrink tends to outperform *rapid*-BoxShrink in the weakly-supervised setting. The difference between the two variants is smaller for bigger models with *rapid*-BoxShrink being once better than *robust*-BoxShrink for the VGG-16 architecture. One explanation could be that bigger models are more robust to the label noise than smaller ones. We want to point out however, that the margin between the two is still overlapped by the standard deviations of both methods.

Future Work. We want to further integrate the framework into the training pipeline by, e.g., adjusting the mean foreground and background embeddings as the model gets better. Also, we have evaluated our approach on a medium-sized data set with binary class segmentation. For a more detailed quality evaluation, an analysis of BoxShrink's performance on multi-class problems and bigger data sets is required. Lastly, starting with BoxShrink pseudo-masks instead of bounding box annotations directly could also improve existing state-of-the-art weakly-supervised learning algorithms.

6 Conclusion

In this work, we presented BoxShrink, a weakly-supervised learning framework for segmentation tasks. We successfully demonstrate the effectiveness of the BoxShrink framework in the weakly-supervised setting on a colonoscopy medical image data set, where we employ bounding-box labeling and output the segmentation masks. Compared to the fully-supervised setting, our weakly-supervised framework shows nearly the same results. Finally, we open-sourced and published the code and bounding boxes for the CVC-Clinic data set .

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: SLIC superpixels compared to state-of-the-art superpixel methods. IEEE Trans. Pattern Anal. Mach. Intell. **34**(11), 2274–2282 (2012)
2. Akbari, M., et al.: Polyp segmentation in colonoscopy images using fully convolutional network. In: 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 69–72. IEEE (2018)
3. Bearman, A., Russakovsky, O., Ferrari, V., Fei-Fei, L.: What's the point: semantic segmentation with point supervision. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9911, pp. 549–565. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46478-7_34
4. Bernal, J., et al.: Comparative validation of polyp detection methods in video colonoscopy: results from the MICCAI 2015 endoscopic vision challenge. IEEE Trans. Med. Imaging **36**(6), 1231–1249 (2017)
5. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 801–818 (2018)
6. Dai, J., He, K., Sun, J.: BoxSup: exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1635–1643 (2015)
7. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255. IEEE (2009)
8. Fan, D.-P., Ji, G.-P., Zhou, T., Chen, G., Fu, H., Shen, J., Shao, L.: PraNet: parallel reverse attention network for polyp segmentation. In: Martel, A.L., Abolmaesumi, P., Stoyanov, D., Mateus, D., Zuluaga, M.A., Zhou, S.K., Racoceanu, D., Joskowicz, L. (eds.) MICCAI 2020. LNCS, vol. 12266, pp. 263–273. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-59725-2_26
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)

10. Huang, Z., Wang, X., Wang, J., Liu, W., Wang, J.: Weakly-supervised semantic segmentation network with deep seeded region growing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7014–7023 (2018)
11. Khoreva, A., Benenson, R., Hosang, J., Hein, M., Schiele, B.: Simple does it: weakly supervised instance and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 876–885 (2017)
12. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
13. Krähenbühl, P., Koltun, V.: Efficient inference in fully connected CRFs with Gaussian edge potentials. In: Advances in neural Information Processing Systems, vol. 24 (2011)
14. Krähenbühl, P., Koltun, V.: Parameter learning and convergent inference for dense random fields. In: International Conference on Machine Learning, pp. 513–521. PMLR (2013)
15. Lin, D., Dai, J., Jia, J., He, K., Sun, J.: Scribblesup: Scribble-supervised convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3159–3167 (2016)
16. Ouali, Y., Hudelot, C., Tami, M.: Semi-supervised semantic segmentation with cross-consistency training. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12674–12684 (2020)
17. Pedregosa, F., et al.: Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
18. Rahman, M.A., Wang, Y.: Optimizing intersection-over-union in deep neural networks for image segmentation. In: Bebis, B., et al. (eds.) ISVC 2016. LNCS, vol. 10072, pp. 234–244. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-50835-1_22
19. Rawat, W., Wang, Z.: Deep convolutional neural networks for image classification: a comprehensive review. Neural Comput. **29**(9), 2352–2449 (2017)
20. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
21. Song, H., Kim, M., Park, D., Shin, Y., Lee, J.G.: Learning from noisy labels with deep neural networks: a survey. IEEE Trans. Neural Netw. Learn. Syst. (2022)
22. Sourati, J., Gholipour, A., Dy, J.G., Tomas-Fernandez, X., Kurugol, S., Warfield, S.K.: Intelligent labeling based on fisher information for medical image segmentation using deep learning. IEEE Trans. Med. Imag. **38**(11), 2642–2653 (2019)
23. Stutz, D., Hermans, A., Leibe, B.: Superpixels: an evaluation of the state-of-the-art. Comput. Vis. Image Underst. **166**, 1–27 (2018)
24. Tang, M., Djelouah, A., Perazzi, F., Boykov, Y., Schroers, C.: Normalized cut loss for weakly-supervised CNN segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1818–1827 (2018)
25. Triggs, B., Verbeek, J.: Scene segmentation with CRFs learned from partially labeled images. In: Advances in Neural Information Processing Systems, vol. 20 (2007)
26. Van der Walt, S., et al.: scikit-image: image processing in python. PeerJ **2**, e453 (2014)
27. Wang, W., Lai, Q., Fu, H., Shen, J., Ling, H., Yang, R.: Salient object detection in the deep learning era: An in-depth survey. IEEE Trans. Pattern Anal. Mach. Intell. (2021)

28. Wei, Y., Feng, J., Liang, X., Cheng, M.M., Zhao, Y., Yan, S.: Object region mining with adversarial erasing: a simple classification to semantic segmentation approach. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1568–1576 (2017)
29. Xing, F.Z., Cambria, E., Huang, W.B., Xu, Y.: Weakly supervised semantic segmentation with superpixel embedding. In: 2016 IEEE International Conference on Image Processing (ICIP), pp. 1269–1273. IEEE (2016)
30. Yang, X., Song, Z., King, I., Xu, Z.: A survey on deep semi-supervised learning. arXiv preprint arXiv:2103.00550 (2021)