Methods for Understanding the Influence of Input Variables on the Decision of a Deep Artificial Neural Network

Master's Thesis of Torsten Dietl
2nd October 2018

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Computer Science Department
Knowledge Engineering Group

Methods for Understanding the Influence of Input Variables on the Decision of a Deep Artificial Neural Network

Submitted Master's Thesis of Torsten Dietl

1. Assessment: Prof. Johannes Fürnkranz
2. Assessment: Dr. Eneldo Loza Mencía

Day of submission:

Abstract

Machine learning (ML) and artificial intelligence (AI) are topics with high industrial impact, due to the enormous success in applying deep artifical neural networks (DANNs) to various pattern recognition tasks. The results of DANNs are so convincing that neural nets are already getting tested in heavily regulated fields like medicine or finance. However, these autonomous systems are deployed without evaluating the reasoning behind the decisions they make. This is due to the fact that despite all the accomplishments DANNs achieved, their decisions are still mostly a black box. Thus, the aim of this thesis is to explain the influence of input variables on the decision of a DANN.

This thesis met its aim through an extensive study of relevant literature and the implementation of practical research. The latter was carried out by developing a new explanation method, called Profiled LICON Analysis (PLAY), and an experiment which compares the proposed method to two existing explanation methods (Linear Weighting Scheme for the Contribution of Input Variables (LICON) and Global Sensitivity Analysis (GSA)). This research has shown the general ability of the PLAY method to accurately calculate the expected influence values on synthetic and real-world data sets. Furthermore, it was proven that the profiling technique used in the PLAY method led to a higher sensitivity compared to the LICON method. This thesis complements the existing scientific knowledge concerning the explanation of DANNs by a categorisation of existing explanation methods and the development and justification of the PLAY method.

Keywords: Deep Artificial Neural Network, Machine Learning, Artificial Intelligence, Explanation, Black Box, Interpretation of Input Influences, Linear Weighting Scheme for the Contribution of Input Variables, Profiled LICON Analysis, Global Sensitivity Analysis

Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

AI	Artificial Intelligence
AUC	Area Under the ROC Curve
CNN	Convolutional Neural Network
DANN	Deep Artificial Neural Network
DARPA	Defense Advanced Research Projects Agency
DCS	Definition Compliance Score
DTD	Deep Taylor Decomposition
GIM	General Influence Measure
GINI	GINI Coefficient
GSA	Global Sensitivity Analysis
IQR	Interquartile Range
IRI	Index of Relative Importance
LICON	Linear Weighting Scheme for the Contribution of Input Variables
LRP	Layer-wise Relevance Propagation
ML	Machine Learning
MSE	Mean Square Error
NSA	Numeric Sensitivity Analysis
OAT	One-At-A-Time
PaD	Partial Derivatives
PaD2	Partial Derivatives 2
PLAY	Profiled LICON Analysis
ROC	Receiver Operating Characteristic

1 Introduction

1.1 Background

Machine learning (ML) and artificial intelligence (AI) are topics with a high industrial impact, due to the enormous success in applying deep artificial neural networks (DANNs) to various pattern-recognition tasks. Some examples of such successful applications can be found in the fields of speech recognition (Hinton et al., 2012), image classification (Krizhevsky, Sutskever, and Hinton, 2012), and reinforcement learning in video games (reaching a level of control similar to humans) (Minh et al., 2015).

The combination of reinforcement learning with DANNs represents another success in the research of AI. For example, the company OpenAI created a bot (computer program) for the video game *Dota 2* which learned strategies to win the game by playing against itself (OpenAI, 2017). More recently, Google DeepMind was able to develop a similar AI program for the game *Go*. By playing against itself, the AI developed strategies for this game exceeding those known to human players and, subsequently, defeated the South Korean grandmaster by 100 to 0 (Sample, 2017).

The results of DANNs are so convincing that neural nets are already being used, in part, to guide physicians' diagnoses, support law firms in advising their clients, and help financial institutions regarding their credit-decision processes (Crawford and Calo, 2016, p. 312). However, these autonomous systems are being deployed without evaluating whether they affect the human population and what this effect might be (ibid., p. 311). Furthermore, despite all the accomplishments DANNs have achieved, their decisions are still mostly a black box. This means, it is not possible to explain the reasoning behind a network's decision. In confronting problems with many input variables and a complex neural network structure, the current research lacks methods to understand the influence of individual input variables on the decision of the neural net.

Efforts were made to solve this lack of understanding mainly by trying to derive decision trees from neural networks and by pruning these trees since it has been a common belief that small systems are easier to grasp (Montavon, Samek, and Müller, 2018). However, this kind of solution has its drawbacks: complex neural networks tend

to generate huge decision trees, and their comprehensibility is questionable. Furthermore, some of the current algorithms for extracting decision trees and rule sets have restrictions. For example, most of the available solutions have focused on discrete data or have needed a specific conversion of the input data (Setiono, Baesens, and Mues, 2008). Thus, researchers have come up with other ways to explain a neural network's decisions such as the information theoretic approach (Papadokonstantakis, Lygeros, and Jacobsson, 2006), fuzzy curves (Lin and Cunningham, 1995), fuzzy rules (Benitez, Castro, and Requena, 1997), and the Euclidian distance (Green et al., 2009). Furthermore, multiple methods have been developed by utilising the connection weights of a neural network (Gevrey, Dimopoulos, and Lek, 2003; Giam and Olden, 2015; Milne, 1995; Olden, Joy, and Death, 2004), by applying sensitivity-analysis approaches (Gevrey, Dimopoulos, and Lek, 2003; Giam and Olden, 2015; Lek et al., 1996; Papadokonstantakis, Lygeros, and Jacobsson, 2006), or by employing the back propagation algorithm (Baehrens et al., 2010; Kasneci and Gottron, 2016; Simonyan, Vedaldi, and Zisserman, 2014; Smilkov et al., 2017). Even though a plethora of possible ways exist to analyse a neural network's decisions, this field of research is still in its infancy, and there is not yet agreement on the method to explain a neural network.

1.2 Research Focus

More and more, autonomous systems are being applied in various fields and are supporting people's daily work. Businesses, companies, and agencies are interested in the possibilities DANNs could provide. For example, the Defense Advanced Research Projects Agency (DARPA) has already expressed its interest in finding 'technology to make this new generation of AI systems explainable' (Defence Advanced Research Projects Agency, 2016, p. 6). However, applying systems such as DANNs without the ability to understand their decisions can be a delicate issue. The need to understand and to explain just why a given neural network selected a specific decision might not appear that urgent or necessary if a neural network recommends, for example, which product to buy or which movie to watch, but it is a different matter to trust DANNs to make medical decisions on their own and to drive cars autonomously. Thus, a way to explain neural network's decisions is desirable.

Trust can only be gained if methods and understandable interpretations that explain which parameters lead to a specific decision are found and agreed upon. As self-driving cars will probably become popular sometime in the future, it will be necessary to explain the decisions made by such cars, for example, in the case of an accident (Firth-Butterfield, 2017). In addition, it would also be valuable to find metrics rating the explanation methods by their statistical reliability. Only if a reliable explanation is accessible, will people be able to trust AIs and to still feel in control. This will allow deep learning to extend its applications in heavily regulated fields such as finances and medicine.

Thus, more research on the explanation of AI's decisions is necessary if the impact of AI systems should continue and extend into these more regulated fields. Therefore, this thesis studies ways to explain and understand DANNs. The implemented research concentrates on static DANNs as the prediction process for a given set of input variables is deterministic in a static DANN. This is helpful since it reduces sources of variance while comparing two methods. Furthermore, this research focuses on the investigation of the input variables' influence on the prediction of a DANN seeing that the influence of a variable is an accepted way to explain other models such as logistic regression, and such an influence shows how important the variable is for the output (Kasneci and Gottron, 2016).

1.3 Overall Research Aim and Indiviual Research Objectives

The 'black-box nature' of DANNs is an issue which needs to be resolved. Therefore, the overall aim of this research is to explain the influence of input variables on the decision of a DANNk. The following subsidiary objectives have been identified of importance to achieve this overall aim:

1. To *define* the term *explanation* with respect to DANNs.
2. To *identify* existing explanation methods.
3. To *develop* a new explanation method for DANNs.
4. To *compare* the proposed method for existing explanation methods.

5. To *assess* the presented methods.

Objective 1 is necessary due to the lack of a commonly accepted definition for the term ‘explanation’ with resepect to DANNs. Furthermore, the definition is then used to rate the methods that are identified for Objective 2 regarding their compliance to the given definition. The resulting ‘definition compliance score (DCS)’ is then used to select methods for the comparison of Objective 4.

Objective 1 and 2 are accomplished in the form of a literature review in Chapter 2. Objective 3 (To develop a new explanation method for DANNs) is the main part of this research and is accomplished by combining the methods identified in order to achieve Objective 2. For Objective 4, experiments are executed to compare existing methods against the method that is developed in this thesis. The result of the comparison is evaluated in Objective 5, and advantages and disadvantages of the proposed method are examined. Details of the research strategy and the experimental setup data are included in Chapter 3.

This thesis adds value to current research in three ways. Firstly, the literature review provides a definition for the term ‘explanation’ and differentiates the terms ‘interpretation’ and ‘understanding’ with respect to DANNs. Furthermore, the literature review gives an up-to-date summary and critical review of several hitherto known methods for explaining neural networks. In doing so, it complements previous examinations, provides a thorough basis for every interested researcher not yet familiar with this topic, and aids communication and discussion. Thirdly, the thesis proposes and evaluates a new method to explain the decisions made by a DANN.

1.4 Short Chapter Outline

Chapter 2, entitled ‘Literature Review’, defines the terms ‘explanation’, ‘interpretation’, and ‘understanding’ with respect to DANNs. In addition, the literature review gives an overview of several existing methods used to assess the influence of an input variable on a network’s output. Altogether, this analysis provides the background for the further research of this dissertation by highlighting the current development of explanation methods.

Chapter 3, ‘Research Methods’, explains the applied research strategy, the general experimental setup, and the reasons why this strategy and this setup were adopted and how they are assessed.

Afterwards, Chapter 4, ‘Description of Methods to Explain the Influence of Input Variables on the Decision of a Deep Artificial Neural Network (DANN)’, describes the newly developed method and the existing methods that were selected for the comparison.

In Chapter 5, ‘An Experimental Evaluation of the GSA, LICON, and PLAY Explanation Methods’, the setup of every experiment is detailed. Furthermore, the results of the experiments are presented, and the results of the newly developed method are compared to the other existing methods.

Chapter 6, ‘Conclusions’, contains an assessment of the new method and its potential issues. Moreover, possible starting points for further research are identified.

2 Literature Review

The study of this review of literature focuses on Objective 1 since it is important to specify a definition for the term 'explanation' in the context of DANNs so as to assess and to compare explanation methods. Furthermore, this literature review addresses Objective 2. In order to do so, this literature review explores current explanation approaches which have been described within academic research, categorises these approaches and highlights challenges and possibilities in them. Both Objective 1 and 2 are relevant to Objective 3.

Overall, studying these areas of literature is meant to provide a meaningful basis for discussion and analysis of the main issue of this thesis – the explanation of DANNs. This section should also provide a thorough basis, moreover, for the subsequent research and should exhibit a comprehensive understanding of the key issues.

2.1 Definition of Explainability

Many of the available research papers agree on the assumption that finding a way to explain DANNs is a crucial task. However, as much as they have agreed on the endeavour of illuminating the 'black box' of DANNs, they have differed in their conception of just how the 'black-box' should be illuminated. This missing consensus is one reason for the various available explanation methods. Achieving an overall understanding and finding a comprehensive explanation for neural networks' decision processes have not always been the key target of scholarship. Previously, the main focus was on the reduction of the complexity of networks through pruning the networks or determining relevant inputs (Engelbrecht, Cloete, and Zurada, 1995; John, Kohavi, and Pfleger, 1994; Reed, 1993; Sung, 1998; Zurada, Malinowski, and Cloete, 1994) since a neural network performs reasonably better with less noisy data (Milne, 1995).

Therefore, John, Kohavi, and Pfleger (1994) have discussed the term 'relevance'. They have pointed out that the hitherto existing four definitions for the term 'relevance' are inaccurate, and they have proposed new definitions for 'relevance', 'weak relevance', and 'irrelevance'. This classification seems reasonable for the pruning task because it allows one to prune irrelevant features. However, it is insufficient when it is used to

explain a neural network because this classification provides no information on the difference between features classified into the same relevance group. Still, this classification could be applied as a part in the definition of 'explainability', if 'explainability' is defined based on input relevance.

The earlier scholarly focus on pruning networks has been further nurtured by the common belief that less complex networks are easier to understand (Montavon, Samek, and Müller, 2018). This means a precise formulation of 'explainability' did not seem to be necessary as pruned networks would become small enough to be intuitively understandable. This led to multiple proposed methods such as the ones by Klimasauskas (1991), Mak and Blanning (1998), and Tchaban, Taylor, and Griffin (1998), which want to 'interpret', 'understand', or 'explain' DANNs without further specifying these terms. Olden and Jackson (2002) were more specific in their task of 'illuminating the "black box"' (ibid., p. 1) as they tried to understand precisely the contributions of variables to the decision in DANNs, which for them meant being able to give an 'interpretation' of the statistical model. They claimed that neural networks are indeed not black boxes anymore and that the existing methods provide enough insights into the inner workings of a DANN to successfully interpret their decision process. The amount of research papers published afterwards, for example Bach et al. (2015), Cao and Qiao (2008), Montano and Palmer (2003), and Montavon et al. (2017), which have proposed new methods and ways to interpret neural network decisions, show that this statement can be questioned. Thus, the black box nature is still not satisfyingly solved even though a substantial number of methods exist which find reasonable correlations between input and output. Most of these methods provide only additional metrics whose interpretation is highly dependent on the modeled problem. In other words, existing methods cannot be applied to new problems without understanding the motives for interpretation as they may produce plausible but misleading explanations. Additionally, the motives of interpretation, and the results of compared methods are diverse and occasionally disagree with each other (Lipton, 2016).

The plethora of terms used in the context of 'explainability' of a DANN and the difference in achieved results dependent on the motives make it necessary to define the terms 'interpretation', 'understanding' and 'explanation', and, therefore, the term 'explainab-

ility' in the context of the analysed method (ibid.). Lipton (ibid.) has approached this problem by evaluating the definition of the term 'interpretability'. Considering that no one has yet defined this term in relation to DANNs, he stated that either 'interpretability is universally agreed upon, [...] or the term interpretability is ill-defined' (ibid., p. 1). In order to resolve the latter, he has given more specific definitions for the term 'interpretability' that are based on the *desiderata* of interpretability research (ibid., p. 2). The five main *desiderata* he has discovered are: 'trust', 'causality', 'transferability', 'informativeness', and 'fair and ethical decision-making'. This means interpretability should serve to either help build trust in a model, to draw causal relations between natural phenomena, to transfer underlying concepts to other domains, to provide additional information besides the output, or to assess whether made decisions conform to ethical standards (ibid., pp. 3–4).

However, these five *desiderata* are partially dependent on each other. In fact, only informativeness can be seen as independent. For example, Lipton has stated that a model may be trustworthy if it mimics a human, that is to say if it only 'tends to make mistakes in regions of the input space where humans also make mistakes' (ibid., p. 3). It is reasonable that models which tend to make mistakes in areas in which a human makes no mistakes are regarded as not trustworthy. However, for a model to be regarded as trustworthy, it is not enough that it makes no mistakes. To trust a decision made by a model, it is necessary to understand the reasoning behind that decision. Therefore, mimicking human decisions has to be combined with informative content to create real trust in cases of complex decisions. Besides trust, the desire to find causality is an understandable aim for many researchers, even though Lipton (ibid., p. 3) has cautioned that these learned associations do not have to reflect causal relationships. However, if one can rely on a model's trustworthiness, it might be reasonable to generate testable hypotheses out of the model's decisions. Thus, the basic purpose of interpretability has to be information extraction. The extracted information could then be put together to achieve higher aims such as trust, ethical decision-making, transferability, and causality.

Lipton (ibid., pp. 4–6), furthermore, has described the properties of interpretable models and their different manifestations. He has broadly divided them into the two main categories of 'transparency' and 'post-hoc interpretability': with transparency con-

cerning everything inside the model itself (that is, understanding how the mechanism of the model works) and with post-hoc interpretability concerning all other information which is outside of the inner mechanism (ibid., p. 4). Furthermore, he has subdivided the main categories into three subcategories for transparency – 'simulatability', 'decomposability', and 'algorithmic transparency' – and four subcategories for the post-hoc interpretability – 'text explanations', 'visualisations', 'local explanations', and 'explanation by example' (ibid., pp. 4–6).

Classifying interpretability as either transparency or post-hoc interpretability is a somewhat rough but reasonable first segmentation. However, while the subdivisions of transparency follow a logical path from a total view (simulatability) to a detailed view (algorithmic transparency) (ibid., p. 4), the subcategories of post-hoc interpretability are a conglomerate of additional information outside the inner mechanism of the model. Due to this fact, such an aligned path cannot be applied to the post-hoc interpretability subcategories, and another classification may be also appropriate – especially as the category 'local explanations' concerns the focus of an explanation, while the other categories concern the representation of the explanation. Nevertheless, this classification model allows researchers to focus their research and to specify a definition of 'interpretability' to validate the developed methods. Furthermore, the research by Lipton (ibid.) has urged the research community to give precise definitions for 'explanation', 'interpretation', and 'understanding'.

A good example for addressing this issue is a paper by Montavon, Samek, and Müller (2018), in which they evaluated methods for 'interpreting and understanding deep neural networks' (ibid.). They were aware of the lack of a coherent definition for the terms 'interpretation', 'understanding', and 'explanation', and they relied on the aforementioned paper by Lipton (2016) so as to define their usage of the term 'interpretation'. Furthermore, they specified 'understanding' in their context and gave a definition for the term 'explanation'. The paper by Montavon, Samek, and Müller (2018) has thus successfully demonstrated how the issue of problem formulation can be tackled.

This thesis follows the example by Montavon, Samek, and Müller (ibid.) and so formulates the problem and the intended goal. Therefore, based on the definitions by Lipton (2016) and by Montavon, Samek, and Müller (2018), the empirical study per-

formed in this thesis focuses on methods regarding post-hoc interpretability and uses the following definitions for the terms ‘understanding’, ‘interpretation’, and ‘explanation’:

Explanation In this thesis, an ‘explanation’ is defined as a collection of concepts which can be interpreted by humans to understand the explained subject. This means the thesis defines an explanation of a DANN as one or multiple methods which answer the following three questions:

1. How is a specific output calculated for a given example?
2. What is the influence of an input on an output for a given example?
3. Why is a specific output the result of a given example?

The first question has already been answered by the calculation formula of a neural net. However, the calculation formula alone is not suitable as an easily understandable explanation. Therefore, the second and third question are more important to answer. The second question tries to aggregate the calculation formula into a single influence value for every input, while the third question wants to find a reason for the respective influence values.

Interpretation An interpretation is defined as the transfer of abstract concepts, that is to say, the results of explanation methods, into a representation that a human can make sense of. The representation should either be generally accepted as understandable (for example, through established charts, height-/heatmaps, and so forth) or be composed of such representations to allow a knowledge transfer.

Understanding The goal of this thesis is to characterise a model’s black-box behaviour, and, therefore, the term ‘understanding’ refers to a ‘functional understanding’ of the model, in contrast to an algorithmic or low-level understanding. For this reason (if a DANN is understood), it should be possible to comprehend the behavior of the DANN and to approximate the model’s output for unknown data based on the interpretation of the results of the explanation methods. Furthermore, there should be an intuitive comprehension of the output value change when a given input is manipulated.

2.2 Existing Explanation Methods

For humans to learn from each other, it is helpful that we are able to describe our decisions and to give reasons for them. Therefore, to further learn from AI and to establish the technology in critical fields such as medicine or law, it is necessary to equip AI with the ability to explain itself. A reasonable way of explaining something is visualising its properties since visualisations are, in general, easier to understand for humans than, for example, numerical data (Rohrer, 2000). Visualising a neural network is commonly done by drawing a directed graph in which every node symbolises a node of the neural net and in which the connections between the graph nodes symbolise the connections between the neural nodes. Besides this architecture, the connection weights are the influential attributes of a neural net. Olden and Jackson (2002) proposed the 'neural interpretation diagram', which displays different thicknesses for the connection lines depending on the weights. This means greater connection-weight values result in a thicker connection line between the nodes. An adaption of this method has used different colors to distinguish positive and negative influences of an input on the predicted output (Tzeng and Ma, 2005).

The neural interpretation diagram and its adaption by Tzeng and Ma are applicable methods for small networks. However, for networks with many nodes, connections, and layers this method is not as suitable since the visualisation becomes too complex to understand. Nevertheless, the underlying premise of developing methods to explain decisions made by classifiers has been understood by other researchers as well. Thus, additional methods were developed to solve this problem by aggregating the data and by calculating a score value. Hereafter, these methods are introduced and categorised into 'connection weight', 'sensitivity analysis' and 'back-propagation-based' methods. Thereby, it is possible to give an overview of the core principal of each category, to describe their differences, to highlight benefits, and to point out limitations regarding their ability to explain DANNs.

2.2.1 Connection Weights

One group of methods is characterised by working only with the connection weights of a neural network. Early methods in this category were mostly adaptions of Garson's algorithm, later called 'product of standardised weights' (Garson, 1991; Gevrey, Dimopoulos, and Lek, 2003; Giam and Olden, 2015; Goh, 1995).

$$Q_{hi} = \frac{|w_{ih} \cdot w_{ho}|}{\sum_{i \in I} |w_{ih} \cdot w_{ho}|} \tag{1}$$

$$R_i = \frac{\sum_{h \in H} Q_{hi}}{\sum_{i \in I} \sum_{h \in H} Q_{hi}} \tag{2}$$

Figure 1: 'Product of standardised weights' R_i for the i-th input variable

The method calculates the relative influence of an input on the output, based on the standardised product of connection weights (the weight of the connection between the input unit and hidden unit and the weight of the connection between the hidden unit and output unit). Milne (1995) proposed a modification of Equation (1) which does not calculate the absolute value of the product of the connection weights. In doing so, the calculated connection weights can indicate a positive or a negative influence. Another adaption of Garson's algorithm was described by Gevrey, Dimopoulos, and Lek (2003), who removed the weight of the connection between the hidden and output neuron from the calculation of Q_{hi}. All three methods are simple in their approach, and Gevrey, Dimopoulos, and Lek (ibid.) were able to further simplify the calculation without significant losses to its meaningfulness compared to Garson's original algorithm. Nevertheless, as these methods were developed on three-layer networks and as they assume the same architecture, they are all limited to such three-layer networks and are, therefore, not applicable to DANNs. Moreover, apart from the approach by Milne, none of the proposed methods is able to distinguish positive and negative relations between input and output variables. However, a distinction between positive and negative relations is more

informative than a ranking of input influence and is, therefore, desired to explain the inner mechanism of neural networks.

Another method derived from Garson's algorithm is the 'product of connection weights' (Olden and Jackson, 2002).

$$R_i = \sum_{h \in H} w_{ih} \cdot w_{ho} \tag{3}$$

Figure 2: The 'product of connection-weights' method by Olden and Jackson (ibid.)

Ibrahim (2013) has introduced a correction factor resulting in the 'modified connection weights', which improved the results even though the improvement is not significant. Better results were produced with Ibrahim's 'most squares' method.

$$R_i = \frac{\sum_{i \in I} (w_{ih}^s - w_{ih}^f)^2}{\sum_{i \in I} \sum_{h \in H} (w_{ih}^s - w_{ih}^f)^2} \tag{4}$$

Figure 3: The 'most squares method' by Ibrahim (ibid.)

In Equation (4), w_{ih}^s represents the initial connection weight (that is, before training) between the input and hidden neuron, and w_{ih}^f is the final weight (that is, after training) of the same connection. While the 'most squares' method is only able to generate a ranking of the inputs' influence on the output value, the 'product of connection weights' method and the 'modified connection weights' method are able to indicate a positive or negative relation between inputs and outputs. However, the 'product of connection weights', the 'modified connection weights', and the 'most squares' method also assume a three-layer network architecture and, therefore, are not applicable to DANNs.

Besides the similarities and obvious differences between Garson's algorithm and the 'product of connection weights' method, the academic world is at odds regarding which of the two methods is the better one. The 'product of connection weights' method, as well as Garson's algorithm, was used and compared in multiple publications (among others, Gevrey, Dimopoulos, and Lek, 2003; Ibrahim, 2013; Kemp, Zaradic, and Hansen,

2007; Montano and Palmer, 2003; Ona and Garrido, 2014; Paliwal and Kumar, 2011; Pentos, 2016). However, Fischer (2015) has come to the conclusion that a substantial number of these realised studies regarding input contribution methods had flaws. For example, the studies were restricted to linear- or semi-linear-related data sets, they used the linear regression as a reference for real data (which does not have to have a linear relation), or they differed in their assessment of the results (ibid.). Therefore, Fisher has re-evaluated the 'product of standardised weights' and the 'product of connection weights' method and has concluded that the 'product of standardised weights' method is superior. Giam and Olden (2015) have doubted Fischer's study and have not been able to confirm its results. As no theoretical basis exists for either the 'product of standardised weights' or the 'product of connection weights' method and as the measured effect is highly dependent on the bias weight, these scholars have deduced that none of the methods are 'entirely faithful approaches to determine variable importance in ANNs' (ibid.).

One method including not only the connection weights but also the bias weights is the 'general influence measure (GIM)' described by Papadokonstantakis, Lygeros, and Jacobsson (2006). It is able to give a relation between the input-specific GIM and the GIM for the bias weights. However, just like the algorithms inspired by Garson's research, it is only applicable to three-layer networks and is not able to indicate a positive or negative relation between input and output values. The same applies to the 'interquartile range (IQR)' method by Paliwal and Kumar (2011). Nevertheless, the IQR method has one notable approach: it calculates the measure using multiple three-layer networks. This increases the computational cost but delivers a more stable result.

Connection weight methods used to be a promising approach for calculating the influence of inputs on the predicted output in times when three-layer networks were most commonly used. One goal of these methods was to prune inputs, which can be done by all of the aforementioned methods. However, except for the method of Milne (1995), connection weight methods are not helpful in supporting the understanding of a neural network's decision seeing that they lack the ability to explain the positive or negative relation between an input and an output. Therefore, methods based solely on

the connection weights have their justifications but cannot be the only approaches for explaining a neural network.

2.2.2 Sensitivity Analysis

Besides the methods using connection weights, another popular way of gaining insight into a neural network is 'sensitivity analysis'. Sensitivity analysis assigns the uncertainty in the output to the uncertainty from different inputs (Saltelli et al., 2008). In other words, sensitivity analysis examines the change of an output in relation to a change of the input. Since this can be measured in multiple ways, multiple directions in the field of sensitivity analysis exist.

One-at-a-Time Methods

One way to measure change in the output is to manipulate an input, to fix all other inputs at a given value, and to compare the generated output to the original one. Therefore, methods using this approach are named 'one-at-a-time (OAT)' sensitivity analyses. Within this field, three general procedures have emerged: building a profile, perturbing the input values, and completely removing input nodes. The advantage of OAT methods is that they can be applied to every model, regardless of its architecture and size; thus, OAT methods are, in general, applicable to DANNs.

'Profile methods' are characterised by dividing the complete input interval of one input into n sub-intervals and by calculating the output for every sub-interval boundary while the other inputs are fixed at certain values. This means that the profile methods approximate the function described by the neural network for the examined input. Therefore, in theory, if the number of sub-intervals approaches infinity, the approximation error converges to zero. However, a high number of sub-intervals result in a high complexity, and, therefore, it is a trade-off between accuracy and computational cost.

The approach by Lek et al. (1996) has set the examined input consecutively to twelve values that are evenly distributed over the complete input interval, while the other input variables are sequentially fixed to the minimum, the first quartile, the median, the third quartile, and the maximum value. For each of the twelve input variations, the median of the resulting five outputs is calculated and then plotted; this gives a visual

indication of the approximated contribution function of the input. Using more than 12 variations for each input value would result in a more exact contribution plot but also in a higher computational expense. Instead of plotting the measurement results, Kewley, Embrechts, and Breneman (2000) have described three methods to aggregate the calculated results to a single measure of the approximated function in their 'one dimensional sensitivity analysis': 'range', 'gradient' and 'variance' sensitivity. They repeatedly calculated the measure over multiple neural networks, which stabilised the results in exchange for increased calculation costs. While the plotted measurements by Lek et al. (1996) allowed them to examine a positive or negative contribution of an input to the predicted output value, this property was lost in the measures by Kewley, Embrechts, and Breneman (2000). Nevertheless, the one-dimensional sensitivity analysis has some justification because it allows one to specify inputs with a high impact on the predicted output, even though one could argue that this could also have been done with the original profile method by Lek et al. (1996).

The 'perturbation methods', such as all OAT methods, change the values of one input node, while they fix the other inputs to their original value. However, instead of dividing the complete input interval into multiple sub-intervals, the perturbation methods define various ways to manipulate the values of the examined input node. Their examination is, therefore, limited to a rather small area around the original value of the examined input node. However, as the aim of a perturbation method, in general, is to measure the impact of small changes on the output (because a large output change for a small input manipulation indicates an extremely sensitive variable), an especially limited application interval is not necessarily a disadvantage if the aim is to get insights into a most specific input-output combination.

Gevrey, Dimopoulos, and Lek (2003) have described the 'perturb' method as a change in the input value from 10% up to 50%, while the other inputs are fixed at their original value. The produced outputs are noted and compared against the original output (for example, by calculating the difference in the mean square error (MSE)). These scholars calculated the difference between the perturbed output and the original output. However, as they were only interested in ranking the input sensitivity, they took the absolute

value of the difference. Therefore, it would be possible to examine the direction of the influence by not taking the absolute value.

Adaptions of the 'perturbation' approach are the 'holdback input randomisation' by Kemp, Zaradic, and Hansen (2007); the 'index of relative importance (IRI)' by Reddy et al. (2015); the 'R^2-based sensitivity analysis' method by Giam and Olden (2015); and the 'mean-based sensitivity analysis' by Pentos (2016). For the 'holdback input randomisation', Kemp, Zaradic, and Hansen (2007) perturbed an input by setting the value of a single input node to an evenly distributed random variable and by calculating the output change on groups of perturbed data records. Reddy et al. (2015) used a +2.5% and a −2.5% change in the input value and calculated the IRI by dividing the difference of the 2.5% changes through the maximum differences of output pairs. The 'R^2-based sensitivity analysis' by Giam and Olden (2015) has calculated the difference between the R^2 measure and the average R^2 measure of m randomly permuted data sets. Pentos (2016) fixed the unexamined inputs to their mean value and calculated the sensitivity of an input by dividing the network error of the perturbed input through the network error of the original input. Despite these multiple perturbation approaches, only the IRI method and an adaption of the 'perturb' method have been able to indicate a positive or negative relation. Moreover, the limitation to an especially local interval around the input value does not necessarily add value if the goal is to explain a DANN. For the local interval, however, the advantage lies in the reduced calculation costs, compared to the profile methods, since a small interval allows a higher density of measurements, given a fixed measurement count.

As mentioned earlier, apart from changing an input value, another way to understand one-at-a-time is to remove the input value completely. Methods using this strategy are the 'classical stepwise' or 'change in MSE' method by Sung (1998), the 'improved stepwise A' and 'B' methods by Gevrey, Dimopoulos, and Lek (2003) and the 'sequential zeroing of variables' method by Papadokonstantakis, Lygeros, and Jacobsson (2006). All these methods share a basic algorithm: they compare a measure, calculated on the original input, against the same measure calculated on an input with one input value removed. The classical stepwise method deletes an input node and re-trains the network to calculate the change in MSE. As this is computationally expensive, the other methods

try to reduce the complexity by zeroing (improved stepwise A and sequential zeroing of variables) the involved connection weight or setting it to its average (improved stepwise B). However, all these methods are not able to indicate a positive or a negative relation between input and output. Furthermore, the classical stepwise method re-trains the network; therefore, the applicability of the result to the original network architecture is not reasonable. Methods removing an input node are thus more useful to prune networks and to find potentially smaller architectures than to explain a given network.

Other mentionable methods using the input data to extract knowledge about a network's inner structure are those proposed by Cortez and Embrechts (2011). They have extended the one-dimensional sensitivity analysis by Kewley, Embrechts, and Breneman (2000), which has resulted in four different variations: the 'Global Sensitivity Analysis (GSA)', the 'data-based sensitivity analysis', the 'Monte-Carlo sensitivity analysis', and the 'cluster-based sensitivity analysis' (Cortez and Embrechts, 2011; Cortez and Embrechts, 2013). It is reasonable to call the GSA method a 'many-at-a-time' method because it is able to calculate the influence of an arbitrary number of input node combinations on the predicted output. The other three methods differ to the aforementioned OAT methods because they use whole sets of data instead of single input vectors to calculate the results. For the data-based sensitivity analysis, the data set is manually picked; for the Monte-Carlo sensitivity analysis, the data set is generated over a uniform distribution. Additionally, for the cluster-based sensitivity analysis, the input interval is divided into evenly distributed chunks, and all input examples are clustered according to the created chunks. Cortez and Embrechts (2011) claimed to have opened the black-box with the GSA method and have proposed visualisation techniques. However, the question as to whether the described techniques really 'explain' a neural network remains unchanged since these two authors give no definition for the term 'explain' nor do they convey what exactly is meant by 'opening the black-box' (ibid., p. 1) .

Montano and Palmer (2003) have described another method, the 'numeric sensitivity analysis (NSA) index', which only works with the data used to train the neural network. It is possible to calculate the standard deviation for the NSA index which can then be interpreted as the behaviour of the function. Thus, a greater standard deviation value indicates a more chaotic or more random function. However, the connection to the

model has been lost, and, therefore, it is not possible to gain insights into the model. As the NSA index examines the underlying function represented by the data, it could be argued that it is, nevertheless, possible to explain a neural network with the NSA index if the error of the network is reasonably low. This can be argued since a low error indicates a good function approximation and, therefore, a negligible difference between the approximated function of the network and the function represented by the data.

In summary, it can be noted that OAT methods, at least most of them, have overcome the absolute value problem and are able to indicate a positive or negative relationship between input and output values, which is an important ability for explaining a neural network. Furthermore, OAT methods are not derived from the structure of the model and only use the model as a black box; this makes these methods applicable to nearly every model, especially DANNs. They, therefore, could be seen as superior to the connection weight methods, despite the fact that their algorithms are not based on the structure of the model. As long as sensitivity methods incorporate the model in some way, the examined results can be related to the model.

Gradient-Based Methods

Another way to measure sensitivity is the usage of the gradients or the partial derivatives. The principle behind gradient-based methods is the usage of the gradient as a contribution measure, even though it is only applicable in a local interval around the examined input value. As these methods work with the gradient, it could be expected that they are always able to indicate a positive or negative relationship between input and output value. However, this is not the case since the sign of the gradient is lost in the more complex calculations. Out of the 'partial derivatives (PaD)' method, the 'partial derivatives 2 (PaD2)' method, the 'input sensitivity' method and the 'first and second order sensitivity analysis' mentioned by Gevrey, Dimopoulos, and Lek (2003), Green et al. (2009), and Yeh and Cheng (2010) respectively, only the PaD method, and the 'first- and second-order sensitivity analysis' are able to indicate a positive or negative relation. However, the other methods have notable approaches. The PaD2 method is able to calculate the partial derivative for a combination of two input nodes, while the input sensitivity method works with neural network ensembles instead of with a single

neural network. Furthermore, the first- and second-order sensitivity analysis examines the linear and the quadratic effect of a three-layer network, which has not previously been done.

In a nutshell, gradient-based sensitivity analysis methods suffer from the same problems as connection-weight methods: most of them are only applicable to three-layer networks and lack the ability to indicate a positive or negative relation. Therefore, they are not useful for explaining DANNs, in general. They do not improve on the connection-weight methods in these aspects. Sung (1998) has tried to overcome the one-hidden layer limitation through the 'two-layer sensitivity analysis'. This method applies to artificial neural networks containing one input layer, two hidden layers and one output layer. However, only the back-propagation methods were able to extend gradients to an arbitrary number of layers and to make the gradient-based approach useable for DANNs.

2.2.3 Back-Propagation-Based Methods

Back-propagation-based methods are characterised by the back-propagation algorithm, which is used to aggregate measures calculated for a single node to a net wide measurement. As the back-propagation principle is applied to all back-propagation methods, these methods only differ in the applied measure. These measures have been categorised by Kindermans et al. (2017a) into three categories: function-, signal- and attribution-based methods.

Function

Function- or gradient-based back-propagation methods calculate a local measure based on the activation function of a node, which is then aggregated. This results in a global measurement of an input's influence based on the local influence in every node. Even though this kind of methods are applicable to DANNs, they share the disadvantage of the previously mentioned gradient-based methods: they are only reliable in a local interval around the examined input values. Therefore, they provide reasonable measurements to explain specific prediction examples, but it is hard to generalise the calculated influences to a general explanation of the assessed neural network.

With regard to a local measure, Baehrens et al. (2010) have developed the local-explanation vectors, which are defined as the probability gradients of a predicted class, while Simonyan, Vedaldi, and Zisserman (2014) have simply used the activation function's partial derivative for their so-called 'saliency map'. Kasneci and Gottron (2016) have gone further and approximated the activation function through a linear function in their 'Linear Weighting Scheme for the Contribution of Input Variables (LICON)' method. Sundararajan, Taly, and Yan (2017) have proposed an additional measure: 'integrated gradients', which are the integral of the partial derivative of the activation function. Despite this variety of local measures, Smilkov et al. (2017) noticed that a lot of existing gradient-based methods were noisy and lacked visual clarity. Thus, they introduced the 'SmoothGrad' method to sharpen gradient-based sensitivity maps. The idea behind this method was to calculate the gradient-based sensitivity for multiple Gaussian-distributed input values around the examined input value and then to use their average as the input sensitivity value. Even though the 'SmoothGrad' sensitivity resulted in a signed value, Smilkov et al. (ibid.) chose to take the absolute value as the sensitivity measure because this value produces a clearer pictures for the 'ImageNet' data set. In doing so, their implementation lost the ability to indicate positive or negative relations, while, in general, the 'SmoothGrad' algorithm is capable of such a distinction.

As mentioned earlier, function- or gradient-based back-propagation methods are an advancement compared to the early gradient-based methods for three-layer networks. Their possible application for DANNs makes them an interesting option, but only if specific data examples are to be examined since they are only reliable in a local interval around the examined point. However, in combination with other methods such as profile methods, it might be possible to solve the local reliability issue and even to expand the usage of the gradient-based back-propagation methods.

Signal

Another form of 'back-propagation sensitivity analysis', inspired by the 'saliency map', is based on the outgoing signal. Signal-based methods were originally developed for convolutional neural networks (CNNs) but their application to other types of neural networks is possible, even though the usefulness of the extracted information on input

data other than images has to be examined. The main principal of signal-base methods is to send the outcome of a prediction (that is, the signal) backwards through the neural network. For CNNs, this can be understood as reconstructing an approximation of the image (the inputs) from the predicted class (the output).

Zeiler and Fergus (2014) have proposed the 'DeConvNet' method, Springenberg et al. (2015) havedescribed the 'guided BackProp' method, and Kindermans et al. (2017a) have developed the 'PatternNet' method. In order to handle the max-pooling layer in a CNN, the 'DeConvNet' method records the switches during a forward pass before running the backward pass, while the 'guided BackProp' method assumes that no max-pooling layer is used. However, Kindermans et al. (ibid.) have shown that the direction of the filter used to remove the distractor does not coincide with the signal direction. They criticised the popular 'DeConvNet' and the 'Guided BackProp' methods because they rely on this false assumption. To solve the issue, Kindermans et al. (ibid.) have described new estimators for the signal which they used in the 'PatternNet' method.

Even though signal-based back-propagation methods originated in working with CNNs, the basic principal (that is, to examine which input pattern causes a given activation) could play a useful part in an explanation algorithm. Nevertheless, the principal of signal-based methods would only explain one part of the decision process; it would not explain the contribution of an individual component of the signal to the output (ibid.).

Attribution

Attribution methods have investigated the contribution of an input to the output. Bach et al. (2015) called this the 'relevance' of an input and proposed the first method to be categorised as an attribution method: the 'layer-wise relevance propagation (LRP)' method. The LRP method assumes that the prediction function $f(x)$ can be decomposed into several sub-functions or layers of computation. Thus, the general idea is to calculate the relevance score for the different computational layers and to define the input's relevance as a composition of the individual sub-relevance scores. Multilayer networks can be decomposed by using the back-propagation algorithm or by using the Taylor-based decomposition described by Bach et al. (ibid.).

Montavon et al. (2017) have extended the work on Taylor-based decompositions and have developed the 'deep Taylor decomposition (DTD)' method, which is a more sophisticated version of the Taylor-type decompositions used in the LRP method. They have also claimed that their method has a more theoretical background than the LRP method. Nevertheless, compared to the LRP method, the implementation by Montavon et al. (ibid.) lacks the ability to indicate positive or negative relations between input and output data. Additionally, it can be difficult to find a root point to develop the Taylor-based decomposition, even though this issue applies to both methods. A solution to the root-point problem has been given by the 'PatternAttribution' method proposed by Kindermans et al. (2017a), which improves the LRP method and can be seen as a root-point estimator for the DTD method, although their method as well lacks the ability to indicate positive or negative relations.

With the attribution, signal, and function methods, the back-propagation sensitivity analysis provides a toolbox for assessing neural networks. Depending on the goal of the investigation, each category has its justification. For a complete explanation of a DANN, it is, therefore, reasonable to expect an examination that uses either one method of each category or that combines them into a single method. Nevertheless, by specialising in DANNs and working with the back-propagation approach, researchers were able to successfully apply different local methods to the complete network. It was possible for them to go beyond the limitations of a three-layer network. Therefore, back-propagation sensitivity analysis is definitively an improvement on the aforementioned connection-weight and gradient-based sensitivity methods for three-layer networks. Additionally, the back-propagation methods combine the network's properties, that is, its architecture and weights, with the data. This makes back-propagation sensitivity analysis one of the most promising approaches for explaining DANNs.

2.3 Emerging Issues and the Need for Empirical Research

A substantial number of methods have been developed to gain insights into the classification decisions of neural networks. The problem could be considered to be solved, and the agreed-upon methods could be used to understand neural networks; however, these assumptions are not accurate. Even though one of the earliest studies began 27 years

ago with Garson's research on the interpretation of connection weights (Garson, 1991), the problem has still not been satisfactorily solved. As the research community still has not agreed on definitions for interpretability, explainability, and understandability, it is difficult to find one true method which can open the neural-net black-box.

To find methods comparable to the method proposed in this thesis, a custom score called the definition compliance score (DCS), is calculated. The DCS evaluates the compliance of a method to the given definitions in Chapter 2.1. It is then calculated for every method and for each of the three definitions (explanation, interpretation, and understanding). The DCS ranges from 0 to 9, with 0 indicating no compliance with the definition and 9 indicating full compliance.

For the definition of explanation, every method is examined through the following questions:

Q1: How is a specific output calculated for a given example?

Q2: What is the influence of an input on an output for a given example?

Q3: Why is a specific output the result of a given example?

The DCS is then calculated by adding up the score points for each question. Score points are assigned by using the following scheme:

3 Points: The question is completely answered by the method.

2 Points: The method gives an extended approach to answer the question.

1 Points: The method gives a basic approach to answer the question.

0 Points: The question is not answered by the method.

For the definition of interpretation, the following rating scheme is applied:

7–9 Points: The method provides an enhanced interpretation approach (for example, by means of colour coding).

4–6 Points: The method gives a simple interpretation approach (for example, by means of a line chart).

1–3 Points: The method only provides calculation results.

The rating scheme for the definition of interpretation does not contain a 0 point rating because even a simple numerical result has a low level of interpretability. Therefore, every method provides some kind of interpretability. For the definition of understanding, it is evaluated if the method enables someone to predict a model's output for unknown data and for manipulated data. Thus, the following rating scheme is applied:

7–9 Points: The method enables the prediction of an output value for unknown input data and of the change in magnitude and direction for manipulated input data.

4–6 Points: The method enables the prediction of the change in magnitude and direction for manipulated input data.

1–3 Points: The method enables the prediction of the change in either magnitude or direction for manipulated input data.

0 Points: The method does not enable any prediction ability.

The resulting scores for the definition of explanation are displayed in Table 1, and the scores for the definitions of interpretation and understanding are displayed in Table 2.

Besides the different DCSs regarding definitions of explanation, interpretation, and understanding, the methods often deviate from each other significantly in their results, are unstable, or are even flawed (Kindermans et al., 2017b). However, it is important not to simply make existing methods more stable. Samek, Wiegand, and Müller (2017) have described why explainability is important in human interactions: on a social level, an explanation for a person's decision or intention helps to build trust; on an educational level, it allows one to build up knowledge, for example, by understanding a teacher's reasoning (ibid.). Even though the social level of interactions between humans and AI is neglectable, the explainability of AI is still important for educational contexts and jurisdiction. Explainability can be used to verify, improve, or learn from the AI system and to make sure the system is compliant to legalisation (ibid.). Therefore, it is necessary to define the goals for AI explainability research well – that is, what and how it should be explained as well as to find stable methods which fulfill these requirements. To summarise, more research in this specific area is necessary in order to establish AI as an accepted technology in our everyday life.

Method	Q1	Q2	Q3	DCS
Neural Interpretation Diagram	1	1	1	3
Adaption of Neural Interpretation Diagram	2	2	2	6
Product of Standardised Weights (and adaptions)	0	1	1	2
Product of Connection Weights (and modification)	0	2	2	4
Most Squares	0	1	0	1
General Influence Measure	0	2	2	4
Interquartile Range	0	1	0	1
Profile	2	3	2	7
One Dimensional	0	2	2	4
Perturbation methods	0	2	1	3
Input removing methods	0	2	1	3
GSA	2	3	3	8
Data-Based, Monte-Carlo and Cluster-Based	2	3	3	8
Partial Derivatives	0	2	2	4
Partial Derivatives 2	0	1	1	2
Input Sensitivity	0	1	1	2
First and Second Order	1	2	2	5
Two-Layer	0	1	1	2
Local Explanation Vectors	0	2	2	4
Saliency Map	0	3	1	4
LICON	1	3	2	5
Integrated gradients	0	3	1	4
SmoothGrad	0	3	1	4
DeConvNet	0	2	2	4
Guided BackProp	0	3	1	4
PatternNet	0	3	2	5
Layer-Wise Relevance Propagation	0	3	2	5
Deep Taylor Decomposition	0	3	2	5
PatternAttribution	0	3	2	5

Table 1: DCS of all methods for the defintion of explanation. Custom illustration based on Bach et al. (2015), Baehrens et al. (2010), Cortez and Embrechts (2011), Cortez and Embrechts (2013), Garson (1991), Gevrey, Dimopoulos, and Lek (2003), Giam and Olden (2015), Green et al. (2009), Ibrahim (2013), Kasneci and Gottron (2016), Kemp, Zaradic, and Hansen (2007), Kewley, Embrechts, and Breneman (2000), Kindermans et al. (2017a), Lek et al. (1996), Milne (1995), Montano and Palmer (2003), Montavon et al. (2017), Olden and Jackson (2002), Olden, Joy, and Death (2004), Paliwal and Kumar (2011), Papadokonstantakis, Lygeros, and Jacobsson (2006), Pentos (2016), Reddy et al. (2015), Simonyan, Vedaldi, and Zisserman (2014), Smilkov et al. (2017), Springenberg et al. (2015), Sundararajan, Taly, and Yan (2017), Sung (1998), Tzeng and Ma (2005), Yeh and Cheng (2010), and Zeiler and Fergus (2014).

Method	Interpretation	Understanding
Neural Interpretation Diagram	8	1
Adaption of Neural Interpretation Diagram	9	2
Product of Standardised Weights (and adaptions)	1	3
Product of Connection Weights (and modification)	2	5
Most Squares	1	0
General Influence Measure	2	3
Interquartile Range	1	0
Profile	6	8
One Dimensional	3	3
Perturbation methods	3	5
Input removing methods	3	3
GSA	7	8
Data-, Monte-Carlo- and Cluster-Based	7	8
Partial Derivatives	2	5
Partial Derivatives 2	1	3
Input Sensitivity	1	3
First and Second Order	2	5
Two-Layer	1	3
Local Explanation Vectors	5	3
Saliency Map	7	4
LICON	4	6
Integrated gradients	7	5
SmoothGrad	7	5
DeConvNet	8	5
Guided BackProp	7	5
PatternNet	7	5
Layer-Wise Relevance Propagation	7	5
Deep Taylor Decomposition	7	5
PatternAttribution	7	5

Table 2: DCS of all methods for the defintion of interpretation and understanding. Custom illustration based on Bach et al. (2015), Baehrens et al. (2010), Cortez and Embrechts (2011), Cortez and Embrechts (2013), Garson (1991), Gevrey, Dimopoulos, and Lek (2003), Giam and Olden (2015), Green et al. (2009), Ibrahim (2013), Kasneci and Gottron (2016), Kemp, Zaradic, and Hansen (2007), Kewley, Embrechts, and Breneman (2000), Kindermans et al. (2017a), Lek et al. (1996), Milne (1995), Montano and Palmer (2003), Montavon et al. (2017), Olden and Jackson (2002), Olden, Joy, and Death (2004), Paliwal and Kumar (2011), Papadokonstantakis, Lygeros, and Jacobsson (2006), Pentos (2016), Reddy et al. (2015), Simonyan, Vedaldi, and Zisserman (2014), Smilkov et al. (2017), Springenberg et al. (2015), Sundararajan, Taly, and Yan (2017), Sung (1998), Tzeng and Ma (2005), Yeh and Cheng (2010), and Zeiler and Fergus (2014).

3 Research Methods

Despite the multiple approaches to explain DANNs, especially extremely large and complex networks still remain a black box for us. This thesis wants to extend the existing research and proposes a new way to illuminate the DANN black box. Therefore, this research has a number of interrelated minor objectives set within the context of explaining DANNs. Objective 1 and 2 have been addressed in the previous chapter by reviewing relevant literature. Even though the review was able to provide useful information about existing explanation methods and about the researchers' opinions regarding the definition of explanation, the review revealed a disagreement in the research community. It is not commonly agreed upon as to how 'explanation' is defined. Furthermore, there is as yet no generally accepted explanation method for DANNs. Even worse, Kindermans et al. (2017b) have shown that some existing methods are unreliable. In order to address this problem, a valuable aspect of this research work relates to Objectives 3, 4, and 5, for which a new explanation method is developed based on existing methods and in compliance with the definitions for explanation, interpretation, and understanding (see in Chapter 2.1).

This thesis aims to justify the proposed method by using experimental research. The newly developed method and two comparative methods are tested with three data sets to determine the method which explains DANNs in the optimal manner (with regards to definitions for explanation, interpretation, and understanding).

3.1 Research Strategy

The overall research strategy that is used to verify the proposed method is experimental research. Experimental research has been defined by Biggam (2015) as an attempt 'to test an hypothesis (i.e. a theory) through some type of experiment'. Scientists use this approach to examine and to test competing hypotheses. The currently prevailing hypothesis is called the null hypothesis, or H_0, and is the subject of the test. Competing against the null hypothesis is the alternative hypothesis, or H_1. If the experiment implemented by the researcher provides evidence that the null hypothesis has to be rejected, then the alternative hypothesis replaces it as the prevailing theory. Using this approach,

scientists are able to exclude erroneous theories and to accept new theories, until they are proven invalid. As this research aims to explain the influence of input variables on the decision of a DANN by developing a new method, it is necessary for this research to justify the newly developed method. Therefore, this thesis implements an experiment to test the following hypotheses:

H_1: The explanation of a DANN given by the proposed method is at least as good and as distinctive as the explanation given by the existing methods.

H_0: The explanation of a DANN given by the proposed method is as good and as distinctive as the explanation given by the existing methods.

Hereby, H_0 is assumed as the prevailing theory because as long as the proposed method is not tested, it is not possible to tell if the method's result can help to explain a DANN better than the existing methods. Furthermore, as the proposed method has not yet undergone scientific validation nor proven its value through studies, like the existing methods have done, the explanation produced by the proposed method is assumed to be not reliable and, therefore, worse compared to the existing methods. Thus, the alternative hypothesis H_1 is a way to justify the proposed method. If the experiment produces evidence that leads to a rejection of the null hypotheses, the alternative hypotheses H_1 has to replace H_0 as the prevailing theory. However, this also means that the explanation of the proposed method is at least as good as the explanation of the existing methods and that it has proven itself.

To gain the necessary data, an experiment was implemented. For the experiment, two existing methods were selected, besides the proposed method. All three methods were used to explain decisions made by a DANN for three different data sets. The experiment was described in further detail in the next section. By assessing and by comparing the resulting data, this research can reject either the null or the alternative hypothesis and, thus, can justify or discard the proposed method.

3.2 General Experimental Setup

As mentioned in Chapter 3.1, the aim of the executed experiments is to justify the newly proposed method by rejecting the null hypothesis.

H_0: The explanation of a DANN given by the proposed method is not at least as good and as distinctive as the explanation given by the existing methods.

Therefore, three experiments were performed to compare the newly developed method with the selected two existing methods. In every experiment, a DANN was trained on a selected data set. Afterwards, each method was used to calculate the input influences for every single input example. Subsequently, the calculated input influences of every method were analysed and compared against each other.

For the experiments, the following three data sets were selected.

- Artificial data
- Real-world data
- Benchmarking data

The artificial data set was generated with given correlations between the input features and the output classes. This was used to assess how far the calculated input influences reproduce the 'real' correlations. To cover the domain of real-world data, the German credit data set by Hofmann (cited in Dua and Karra Taniskidou, 2017) was selected. The data set was choosen because it mimics the data of a heavily regulated field which requires reliable explanation methods to be able to apply DANN in the credit decision process. It consists of 20 input features and one binary output, classifying customers into the categories 'good credit risk' (1) and 'bad credit risk' (2). The input features contain categorical data such as personal status and sex as well as numerical data such as credit duration in months and age in years. Furthermore, this data set can be used by simpler and easier models such as logistic regression. Therefore, it is possible to compare the results of the explanation methods to already accepted explanations.

Lastly, the MNIST database of handwritten digits was selected as the benchmark data set (Y. LeCun and Burges, 1998). It is a commonly accepted and applied data set in the research field of DANNs (Keysers, 2007; Mizukami et al., 2010; Simard, Steinkraus, and Platt, 2003). The classification task is easily solvable for humans, and, thus, the calculated influences can be assessed for being plausible by visual inspection.

In every experiment, the evaluated methods worked with preprocessed data. Thus, especially discrete input variables were 'split' into multiple input values, creating an input

neuron for each possible manifestation of the discrete variable. Additionally, an input neuron was created for an unknown manifestation of the discrete input variable. Furthermore, this thesis limited the experiments to binary classifiers and, thus, to DANNs containing a softmax layer with two neurons as the output layer. However, there was no limitation in the application of the methods since each multi-class problem can be solved by multiple binary classifiers working in a one-versus-all manner.

3.3 Data Analysis

To assess the examined methods, it is necessary to analyse the produced 'input-influence-data'. As the experiments only used binary classifiers, the calculated influence values are interpreted as supporting the target class if the influence value is positive, and they are interpreted as rejecting the target class if the influence value is negative. A variable's influence may be positive and negative at the same time; this indicates a support of the target class (for some values of the input variable) as well as a rejection of the target class (for other values of the input variable). Therefore, the manifestations of the influence values of a pre-processed, discrete input variable can be aggregated by adding all positive values of the manifestations as the variable's positive influence and by adding all negative values of the manifestations as the variable's negative influence.

The analysis of the calculated influences focused on two major examinations. First off, it was assessed whether the proposed method is valid. Secondly, it was evaluated if influences calculated with the new method are more accurate than the two existing methods. To test if the proposed method is valid, it was examined whether the new method resulted in influence values that are similar to influences produced by the commonly accepted models. For the artificial data set and the German credit data set the logistic regression model was used as a benchmark. Regarding the MNIST database of handwritten digits, it was evaluated whether the new method provided influences which reflect the human intuition for classifying the digits. For the second examination, evaluating whether the new method produced superior influence values, the calculated influences of the three methods for all data sets were compared. As the real correlation values were known for the artificial data set, it was possible to see which method produced influences closer to the existing impact of the input variables. The same pro-

cess was applied to the German credit data set, for which the correlation coefficients of the logistic regression model were seen as the real correlation values. For the MNIST data set, the comparison was again done by visual inspection to find the method which produces the most reasonable influences.

Additionally, the selectivity of each method was compared. This means that for every method a data set was created containing the calculated influences and the original class label. Afterwards, a logistic regression model was trained for which metrics were calculated to evaluate whether the newly proposed method is more selective than the existing methods.

The following metrics were used, where applicable, to evaluate the methods:

Precision: The precision is a percentage metric which describes how many of the samples classified as the target class ($truePositive + falsePositive$) were correctly classified as the target class ($truePositive$). Thus, precision is defined as $Precision = \frac{truePositive}{(truePositive+falsePositive)}$ (Sokolova, Japkowicz, and Szpakowicz, 2006).

Recall: The recall is a percentage metric which describes how many of the expected target class samples ($truePositive + falseNegatives$) were correctly retrieved ($truePositive$). Thus, recall is defined as $Recall = \frac{truePositive}{(truePositive+falseNegatives)}$ (ibid.).

F-Measure: The f-measure is a harmonic mean of precision and recall. In this thesis, the F_1 score is used, which is defined as $F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision+Recall}$ (ibid.).

Accuracy: The accuracy measure is a percentage metric which defines how well a binary classifier correctly identifies the classes. This means how many samples of all samples ($truePositive + falsePositive + trueNegatives + falseNegatives$) are correctly classified ($truePositive + falsePositive$). Thus, accuracy is defined as $Accuracy = \frac{(truePositive+trueNegatives)}{(truePositive+falsePositive+trueNegatives+falseNegatives)}$ (ibid.).

AUC: The area under the ROC curve (AUC) measure is used as a heuristic to assess the performance of classification models. An area under the ROC curve (AUC) value of 1.0 indicates perfect classification accuracy, and an AUC value of 0.5 describes the accuracy of a random classification. Furthermore, the AUC measure indicates the

selectivity of a classification model. The nearer the AUC value is to 1.0, the more selective the model is (Schulte-Mattler, Daun, and Manns, 2004).

GINI: The GINI coefficient (GINI) is used to describe how close a classification model is to the abilities of a clairvoyant. A GINI coefficient (GINI) value of 1.0 indicates perfect clairvoyant abilities, while a GINI value of 0.0 describes a model equal to a random process. In the applied statistics, usual GINI values for single key figures are between 0.2 and 0.4. More complex models using multiple features are able to reach GINI values between 0.6 and 0.7. Like the AUC the GINI is used as a measure for the selectivity of a model (ibid.).

Besides the analysis of the produced influence values, it was evaluated whether the proposed method is at least as good as the comparing methods regarding their DCS. In order to do so, the compliance of the newly developed method to the definitions of explanation, interpretation, and understanding (see Chapter 2.1) was assessed and motivated. Based on the described analysis, this thesis was able to reject either the null hypothesis – and thus justify the proposed method – or the alternative hypothesis – and thus discard the proposed method.

3.4 Limitations and Potential Issues

Every research strategy has its advantages and disadvantages. For experimental research, one issue is sample size (Biggam, 2015). Without a large enough sample size, it is not possible to make statistically significant generalisations. However, for a single test concerning one of the selected data sets, this issue is covered by the data sets themselves since they contain at least 1,000 records. Therefore, the influences calculated for all records were statistically justified.

Another concern Biggam (ibid.) has mentioned regarding the sample selection is that the sample data must be selected in a way which allows generalisation. For this thesis, this concern was taken into consideration by covering multiple domains during the selection of the data sets for the experiments. However, as only three data sets were selected, a generalisation might not have be statistically significant and might only have

applied to the three domains. Therefore, further experiments in other domains might be necessary to fully solve this limitation.

The performed experiment was prepared thoroughly and conducted with the aim to obtain valid and reliable data. However, experimental research only becomes trustworthy by repetition. Therefore, this thesis intends to provide a base for validation and further research by giving a detailed description of the used data sets, the applied preprocessing techniques, the architecture of the used DANNs, and the tested methods.

4 Description of Methods to Explain the Influence of Input Variables on the Decision of a Deep Artificial Neural Network (DANN)

As shown in the literature review in Chapter 2.2, multiple methods have been proposed to determine the influence an input has on the output calculated by a DANN. Popular approaches include sensitivity analysis and the back-propagation-based methods.

The method proposed by this thesis is compared against the LICON method by Kasneci and Gottron (2016) and the GSA by Cortez and Embrechts (2011). The former belongs to the group of back-propagation-based methods and uses gradients to calculate the influence of an input on the output; the latter is a profiled sensitivity analysis. This thesis has developed a new method, based on the LICON method. The proposed method has been called Profiled LICON Analysis (PLAY) and enhances LICON through the profiling features of the GSA method.

Subsequently, the methods applied in this experimental research are detailed. For a better understanding of the PLAY method, the description of the methods starts with the LICON and GSA method. Furthermore, symbols used in the description or in the given algorithms are explained in Table 15 of Appendix A, 'Overview and Use of Notation'.

4.1 Linear Weighting Scheme for the Contribution of Input Variables (LICON)

Kasneci and Gottron (2016) developed the LICON method 'inspired by the variable weighting scheme in the log-linear combination of variables in logistic-regression'. Their method has shown that the function represented by a DANN can be successfully approximated in a local neighbourhood of the input values using LICON. The method approximates the local behaviour of a neuron. This local behaviour is then aggregated for all neurons to a DANN-wide influence behaviour. For the explanation of neuron local behaviour, Kasneci and Gottron (ibid.) used gradients (which are easy to calculate) since the usual design of neural networks (using back propagation as a training method) involve derivable activation functions.

The LICON algorithm implemented in this thesis, as one can see in Algorithm 1, calculates the weighted gradients of the local linear approximations $\beta_{ij}^{(l)}$ for every neuron $z_i^{(l)}$ in every layer, and it aggregates them to the network-wide influence values $\alpha_{ik}^{(\#L)}$

for the i-th input value $x_i^{(0)}$ on the k-th output value. The algorithm starts by initialising the influences for the input variables. To do so, it calculates Kroenecker's delta, which can be seen as an identity matrix of the size $\#I \times \#I$. Thus, the method assumes that every input variable has an influence of 1 on itself and an influence of 0 on any other variable.

Algorithm 1 LICON algorithm as implemented for this thesis.

function LICON($M, x^{(0)}$)

$$\alpha_{ik}^{(0)} \leftarrow \begin{cases} 1 & \text{if } k = i \\ 0 & \text{if } k != i \end{cases} \quad \forall i, k = 1, ..., \#I$$

for ($l = 1 \ldots \#L$) **do**
  $\vec{\beta^{(l)}} \leftarrow CalculateGradients(x^{(l-1)}, w^l, b^{(l)})$
  for ($k = 1 \ldots \#N^l$) **do**
    for ($i = 1 \ldots \#N^0$) **do**
      $sum \leftarrow 0$
      for ($j = 1 \ldots \#N^{l-1}$) **do**
        $sum \leftarrow sum + \beta_{kj}^{(l)} * \alpha_{ij}^{(l-1)}$
      end for
      $\alpha_{ik}^{(l)} \leftarrow sum$
    end for
  end for
end for
return $\alpha^{(\#L)}$
end function

In the next step, the algorithm iterates over every layer to calculate the aggregated influences for every input value on the output of the examined layer. To calculate the aggregated influence of the i-th input value, the algorithm sums up the product of all 'incoming' influence values of the previous layer ($\alpha_{ij}^{(l-1)}$) with their associated weighted gradients ($\beta_{kj}^{(l)}$). As one can see in Algorithm 2, a weighted gradient $\beta_{ij}^{(l)}$ is calculated by multiplying the weight $w_{ji}^{(l)}$ with the derivative of the activation function of neuron $z_i^{(l)}$ evaluated for the sum of weighted inputs feeding into neuron $z_i^{(l)}$.

Thus, LICON calculates the influence of an input variable for the first layer and feeds it through every layer so as to get the overall influence of an input variable on a specific output value. In the evaluation of the resulting data, only the influence values for the target class output were used. This was possible because this thesis is limited to binary

Algorithm 2 Algorithm to calculate the gradients for the LICON method.

function CALCULATEGRADIENTS($x^{(l-1)}, w^l, b^{(l)}$)
  for ($i = 1 \ldots \#N^l$) **do**
    for ($j = 1 \ldots \#N^{(l-1)}$) **do**
      $val \leftarrow \sum_{k=1}^{\#N^{(l-1)}} (w_{ji}^{(l)} \cdot x_j^{(l-1)}) + b_i^l$
      $\beta_{ij} \leftarrow w_{ji}^l \cdot f_i^{(l)\prime}(val)$
    end for
  end for

  return β
end function

classifiers which were using a softmax layer with two neurons as the output layer, and, hence, the influence values of the non-target class are just the negated influence values of the target class and vice versa. Furthermore, all influence values of discrete input variables were aggregated by adding all positive influences into one influence value of the discrete input variable supporting the target class, and all negative influence values were added into one influence value rejecting the target class.

4.2 Global Sensitivity Analysis (GSA)

Cortez and Embrechts (2011) have developed the GSA method as a generalisation of the work done by Kewley, Embrechts, and Breneman (2000) and by Embrechts, Ozdemir, and Kewley (2003). The GSA method is a sensitivity analysis using profiled inputs to measure the influence of an input variable or a combination of input variables. That means that the method generates input samples based on a base-input vector. Then, the evaluated DANN is used to predict the output values for each of the generated input samples. Afterwards, the algorithm aggregates the predictions by using a sensitivity measure to calculate the influence for the examined input variable(s).

The algorithm implemented in this thesis differs slightly from the algorithm described by Cortez and Embrechts (2011) in two respects. Firstly, this thesis used a given input example as the base-input vector instead of an input vector consisting of the mean or median values for every input variable. This was conducted because the definition of explanation, given in Chapter 2.1, asks (among other things) for the influence of an input on an output for a given example. Furthermore, LICON is already able to calculate

the influence values for a given input example; therefore, in terms of comparability, it is useful to adjust the GSA method in this regard. Secondly, in parallel to the LICON method, the calculated influence value for the non-target class was interpreted as a negative influence for the input variable on the target class. This decision was made because the thesis is limited to binary classifiers. Hence, a rejection of the target class is equal to a support of the non-target class and vice versa.

Algorithm 3 GSA algorithm as implemented for this thesis.

function GSA($M, F, p, x_b^{(0)}$)
  $X \leftarrow CreateProfiledInputs(F, p, x_b^{(0)})$
  for ($a \in F$) **do**
    for ($h = 1 \ldots \#X$) **do**
      $\vec{y}^{(h)} \leftarrow Predict(M, X[h])$
    end for

    $g_{a0} \leftarrow \sum_{h=2}^{\#X} |y_0^{(h)} - y_0^{(h-1)}| / (\#X - 1)$
    $g_{a1} \leftarrow \sum_{h=2}^{\#X} |y_1^{(h)} - y_1^{(h-1)}| / (\#X - 1)$
  end for

  return g
end function

So, as one can see in Algorithm 3, the core concept of the GSA algorithm is to predict the output vector for multiple inputs and to aggregate them by using a sensitivity measure. Cortez and Embrechts (ibid.) have described three different possible sensitivity measures: range, gradient, and variance. For this thesis, the gradient metric was used because of its similarity to the LICON approach, which is based on gradients, as well.

The profiling of inputs plays a large part in the GSA method. Cortez and Embrechts (ibid.) came up with a way to create profiles for combined inputs. Their approach was implemented in this thesis, as one can see in Algorithm 4. At first, profiles were created separately for each input variable. For discrete features, this was done by creating a list containing every possible option for the examined input variable a. Continuous features were then profiled by using a regular sequence. This means that the created list of possible values contained the minimum and maximum value of the input variable

Algorithm 4 Algorithm used to profile a single input variable or a combination of input variables.

function CREATEPROFILEDINPUTS($F, p, x_b^{(0)}$)
  for ($a \in F$) **do**
    if (a is a discrete feature) **then**
      for ($i = 1 \ldots (\#opt(a))$) **do**
        $S_a[i] \leftarrow opt_i(a)$
      end for
    else
      $k \leftarrow \frac{max(a) - min(a)}{(p-1)}$
      $S_a[1] \leftarrow min(a)$
      for ($i = 2 \ldots (p-1)$) **do**
        $S_a[i] \leftarrow k \cdot S_a[i-1]$
      end for
      $S_a[p] \leftarrow max(a)$
    end if
  end for

  $t \leftarrow 1$
  for ($a \in F$) **do**
    $t \leftarrow t \cdot \#S_a$
  end for
  $e \leftarrow 1$
  for ($a \in F$) **do**
    $m \leftarrow t/(e \cdot \#S_a)$
    $R_a \leftarrow Repeat(S_a, e, m)$
    $e \leftarrow e \cdot \#S_a$
  end for

  for ($i = 1 \ldots t$) **do**
    $x^{(0)} \leftarrow [\,]$
    for ($a \notin F$) **do**
      $x_a^{(0)} \leftarrow x_{ba}^{(0)}$
    end for
    for ($a \in F$) **do**
      $x_a^{(0)} \leftarrow R_a[i]$
    end for
    $X[*] \leftarrow x^{(0)}$ ▷ Add the input vector $x^{(0)}$ to the list of profiled inputs X
  end for

  return X
end function

a as well as $p-2$ equally distributed elements between the minimum and maximum value, sorted in an ascending order.

Algorithm 5 Algorithm used to repeat a partial input when combining multiple features in the profiling process.

```
function REPEAT(S_a, e, m)
    r ← []
    for (i = 1 ... m) do
        for (j = 1 ... #S_a) do
            for (k = 1 ... e) do
                r[*] ← {S_a[j]}          ▷ Add the value S_a[j] to the list r
            end for
        end for
    end for

    return r
end function
```

Afterwards, these separate profiles were repeated for the total number of combinations, as one can see in Algorithm 5. This ensured that each possible combination of input values existed. For this, the *Repeat* method repeated m times the list of profiled values for an input variable a, while repeating e times each element of a. By calculating m and e dynamically, based on the already repeated input profiles, it was guaranteed that every possible combination of values for combined input variables was produced.

To give an example, let the profiled inputs for input variable a_1 be $\{1,2\}$, and let the profiled inputs for input variable a_2 be $\{3,4\}$. Then, the total size of possible combinations is $t = 4$, the repetition of each element in the first step is $e = 1$, and the repetition of the whole profile is $m = 4/(e \cdot \#S_{a_1}) = 2$. Therefore, the repeated list for a_1 is $\{1,2,1,2\}$. In the next step $e = e \cdot \#S_{a_1} = 2$ and $m = 4/(e \cdot \#S_{a_2}) = 1$. Therefore, the repeated list for a_2 is $\{3,3,4,4\}$.

Finally, the input vectors are composed of the repeated profile lists. This was done by using the values of the repeated profiles for the profiled input variables $a \in F$ and the values of the base vector $x_b^{(0)}$ for non-profiled input variables $a \notin F$. The input vectors thus produced were added to the list of profiled input vectors X. For every input vector $x^{(0}$ in X, the predicted output was calculated and used in the sensitivity measure, which is the influence of an input variable a on the target class.

4.3 Profiled LICON Analysis (PLAY)

The newly proposed method 'Profiled LICON Analysis (PLAY)' is a combination of the two aforementioned methods, LICON and GSA. In fact, it can be seen as an enhancement of the LICON method through the input-profiling mechanism of the GSA method. Another way to interpret the PLAY method is as a LICON-based sensitivity measure for the GSA method.

Algorithm 6 PLAY algorithm as implemented for this thesis.

function PLAY($M, F, p, x^{(0)}$)
  $X \leftarrow CreateProfiledInputs(F, p, x^{(0)})$
  for ($h = 1 \ldots \#X$) **do**
    $A[h] \leftarrow LICON(M, X[h])$
  end for
  for $i = 1 \ldots \#I$ **do**
    $m_{i0} \leftarrow A_{[}*, \bar{i}, 0]$
    $v_{i0} \leftarrow \sum_{h=0}^{\#X} (A[h, i, 0] - m_{i0})^2 / \#X$
  end for

  return $\{m, v\}$
end function

As one can see in Algorithm 6, the PLAY method starts just like the GSA method by creating profiled inputs for the combination of evaluated input variables F and is based on the base-input vector $x^{(0)}$, which is the examined input vector. Afterwards, the LICON approach is applied to every input $X[h]$ generated in the profiling process. The subsequently produced LICON influences are stored in the matrix $A^{\#I \times \#X \times \#Y}$. By using the stored influences, the mean m_{i0} and the variance v_{i0} of the influence of an input value on the target class ($\vec{y}_0^{(h)}$) are calculated. Mean and variance are processed 'column-wise', which means that they are calculated for a given input value x_i^0 over all LICON influences in A.

Like the LICON approach, only the influence values for the target class output are used in the evaluation of the resulting data because only binary classifiers are used. Furthermore, like the LICON method, all influence values of discrete input variables are aggregated by adding positive influences and negative influences separately for the positive or negative influence of a discrete input variable on the target class.

5 An Experimental Evaluation of the GSA, LICON, and PLAY Explanation Methods

The literature review presented in chapter 2 defined of the terms explanation, interpretation and understanding Furthermore, it provided an overview of existing methods that claim to explain the influence of input variables on the decision of a DANN. In the previous chapter 'Description of Methods to Explain the Influence of Input Variables on the Decision of a Deep Artificial Neural Network (DANN)' three explanation methods were described in detail. In this section these three methods are evaluated and their results are compared.

To do so, three DANNs were trained using a 10-fold cross validation on three different data sets: an artificial data set, the German credit data set and the MNIST database of handwritten digits. Subsequently, each method was applied to the trained network for chosen inputs of the training data set. For the GSA and PLAY methods (as they are profiled methods) six profile points, equally distributed over the complete input interval, were used to calculate the influence values for each of the variables. In this way, every variable was profiled independently while the other variables were fixed to their value in the base input vector. The resulting influences were then assessed to identify the methods which calculate the influences most accurately.

For the artificial data set, influences were produced for each record of the training data set. The mean and variance were then calculated for every input variable over all the influence records. As the artificial data set provides a given correlation between the input variables and the output, it was easy to determine the method which calculated the influences the most accurately. The the same process was applied for the German credit data set. However, instead of a predefined correlation between the variables, the calculated influences were compared to the correlation coefficients of a linear regression applied on the German credit data set. This was done because the German credit data set can be solved by linear regression, which is a commonly used and accepted method in practice (Ploeg, 2010). Therefore, the influences by the linear regression were considered as 'truth' for this real-world data set. As with the approach of the artificial data set, the method with results complying most closely to the correlation coefficients of the

linear regression was accepted as calculating the best influences. Next, for the MNIST data set chosen input samples were used to calculate the influence values for each method. The resulting influences were displayed as images for visual inspection, as performed by Kasneci and Gottron (2016). In the resulting images positive or supporting influences were coloured in blue, while negative or rejecting influences were displayed in red. This procedure enabled analysis of how well the influences calculated by the explanation methods reflect human intuition.

In a final step, each method was used to create an 'influence data set'. This was done by applying every method to each record of the training data sets. The resulting influences for every record of the training data sets then built the influence data set for each method and each data set. Thus, the used influence data set consisted of two variables (positive influence and negative influence) for each variable in the original data set. Furthermore, the influence data set contained the mean value and the variance for every individual input feeding into the associated neural network. It should be mentioned that multiple mean and variance variables existed for one input variable of the original data set. This occurred due to the fact that discrete input variables are one-hot encoded, and therefore the mean and variance values were calculated for each manifestation of the discrete input variable. These influence data sets were then utilised to train a logistic regression, using a 10-fold cross validation, with the aim to determine the most selective method. To assess the selectivity, the GINI and AUC measures for the individual logistic regression models were then compared.

The training of the DANNs and the logistic regression model and the calculation of the analysed measures were performed in a proprietary software tool created by a medium-sized German company, which does not wish to be named. Calculation of the correlation coefficients in the logistic regression was carried out using the 'Weka 3 Data Mining Software' (Frank, Hall, and Witten, 2016). Every other evaluation, for example the calculation of influences using the described methods, was conducted using self-implemented code.

5.1 Artificial Data Set

5.1.1 Data Set

The theoretical ability of the examined methods to calculate influences according to existing correlations is first evaluated. To this end, an artificial data set with predefined correlations between the dependent class variable and the input variables was used to calculate the input influences. The data set consisted of one binary class variable y and five input variables $x_{0.8}, x_{0.6}, x_{0.4}, x_{0.2}$ and $x_{0.0}$. Each of the five input variables x_p was defined to be correlated to y by p, as shown in Figure 4.

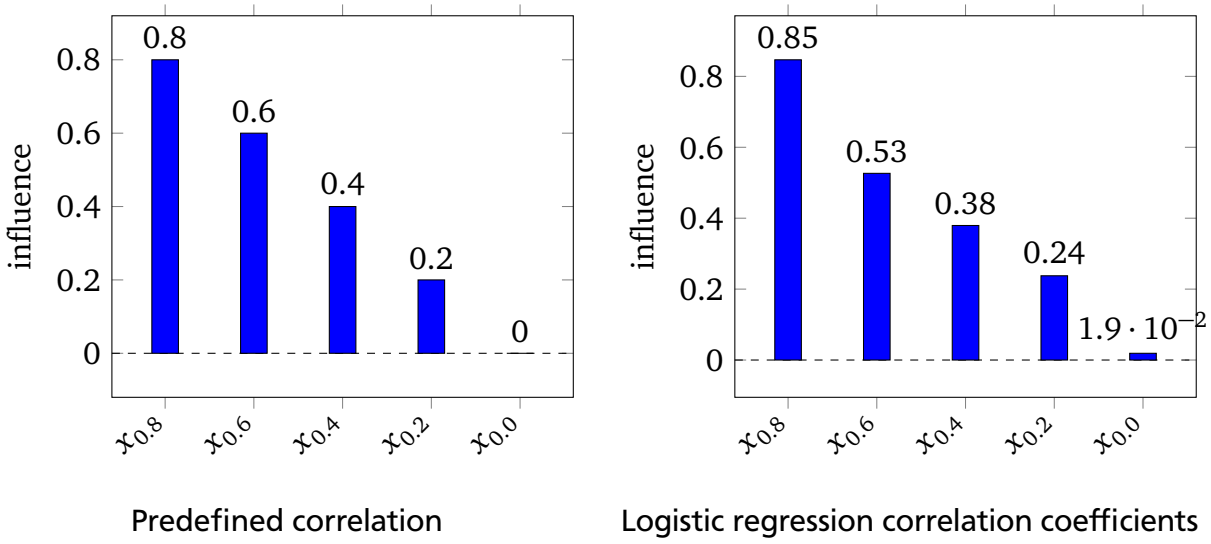


Figure 4: Predefined correlation between x_p and y and the calculated correlation coefficients using a logistic regression for the artificial data set.

With these predefined correlations, a data set of 1,000 records was generated by utilising algorithm 7 to randomly produce the input variables and the dependent class variable for each record. As the records were randomly generated, the minor deviation occuring in the calculated correlation coefficients was expected and is considered reasonable. Nevertheless, it must be guaranteed that the correlation of the generated data is sufficiently similar to the predefined correlation to be useful. To ensure the predefined correlations in the randomly generated data set, a logistic regression was applied and the correlation coefficients were calculated. As seen in Figure 4, the need for similarity between of the resulting correlation coefficients and the predefined correlation values is satisfied.

Algorithm 7 Algorithm to generate an artificial data set with predefined correlations.

function GENERATEARTIFICIALDATASETRECORD
  $\zeta \leftarrow 0$
  for $p \in \{0.8, 0.6, 0.4, 0.2, 0.0\}$ **do**
    $x_p \leftarrow NextRandomGaussian()$
    $\zeta \leftarrow \zeta + p \cdot x_p$
  end for
  $prob \leftarrow 1/(1 + e^{-\zeta})$
  $y \leftarrow BernoulliDistributionSample(prob)$
  return $\{y, x_{0.8}, x_{0.6}, x_{0.4}, x_{0.2}, x_{0.0}\}$
end function

As seen in algorithm 7, the input variables are sampled over a standard normal distribution with a mean of 0 and a variance of 1 (indicated by the method call *NextRandomGaussian()*) and the dependent class variable y is drawn from a Bernoulli distribution (indicated by the method call *BernoulliDistributionSample(prob)*). The probability *prob*, applied to the Bernoulli distribution, is the result of the inverse logit function evaluated in ζ. ζ is calculated as the sum of all input variables x_p weighted with their associated predefined correlation value p. Therefore, the values of the input variables are set mostly between -3 and 3, while the class variable y is a discrete variable with values 1 and 0, of which 1 marks the target class.

5.1.2 Neural Network Architecture

A neural network was created for the evaluation of the methods with the artificial data set. The architecture of the network contained one input layer, one hidden layer and one output layer. In the hidden layer, five neurons were applied using the logistic function as the activation function. The output layer was built from two neurons using the softmax function for activation.

After 40 iterations on the full artificial data set the training was stopped because the model did not improve further. This is shown in appendix F, 'Model Adjustments for the DANN Models Trained on the Artificial and German Credit Data Sets'. The resulting Receiver Operating Characteristic (ROC) curves can be seen in appendix B, 'ROC Curves for the Artificial Data Set'.

Measure	Target class	Non target class	Weighted mean
AUC	0.7428524749286821	0.7428524749286821	0.7428524749286821
GINI	0.48570494985736407	0.48570494985736407	0.48570494985736407

Table 3: Measured results for the 10-fold cross validation training of the DANN model used on the artificial data set.

Measure	Target class	Non target class
Precision	0.7441860465116279	0.7368421052631579
Recall	0.6808510638297872	0.7924528301886793
F-Measure	0.711111111111111	0.7636363636363637
Accuracy	0.74	0.74
AUC	0.8161380971497395	0.8161380971497392
GINI	0.632276194299479	0.6322761942994783

Table 4: Measured results for the selected DANN model and a decision threshold of 0.5 used on the artificial data set.

Table 3 shows the AUC and GINI values achieved following the 10-fold cross validation training. The resulting weighted mean AUC value of ~ 0.743 and the GINI value of ~ 0.486 indicate that the used network architecture is capable of successfully predicting classes based on the artificial data set. Furthermore, the AUC value over all model instances of the 10-fold cross validation (~ 0.743) is similar to the AUC value of the logistic regression model (~ 0.749), which was used to guarantee the correlations between the input variables and the output variable. Additionally, the GINI value of ~ 0.486 confirms an expected degree of selectivity (Schulte-Mattler, Daun, and Manns, 2004).

For the experiment, the model with the highest AUC, GINI, accuracy and f-measure was selected. As presented in Table 4, the selected model achieved a precision of ~ 0.744, a recall of ~ 0.681, an f-measure of ~ 0.711, an accuracy of ~ 0.74 and an AUC value of ~ 0.816 for the target class, and reached similar values for the non-target class. For comparison, the logistic regression model, which was used to guarantee the correlations between the input variables and the output variable, achieved a precision of ~ 0.69, a recall of ~ 0.653, an f-measure of ~ 0.671, an accuracy of ~ 0.686 and an AUC value of ~ 0.749 for the target class. The neural network model is therefore superior to the used logistic regression model. Additionally, the achieved GINI value of ~ 0.632 is comparable to accepted GINI values in the rating industry (ibid.). Hence, it

has been shown that the trained DANN model is eligible for use with the artificial data set.

5.1.3 Experimental Results

Comparison of mean influence values and variance

The experiment using the artificial data set was performed with the aim to verify the theoretical ability of the three methods to calculate the influence of an input variable on the output variable. As the artificial data set was constructed with predefined correlations between the input variables and the output variable, an explanation method was considered to accurately calculate the influences if the predefined correlation values were returned.

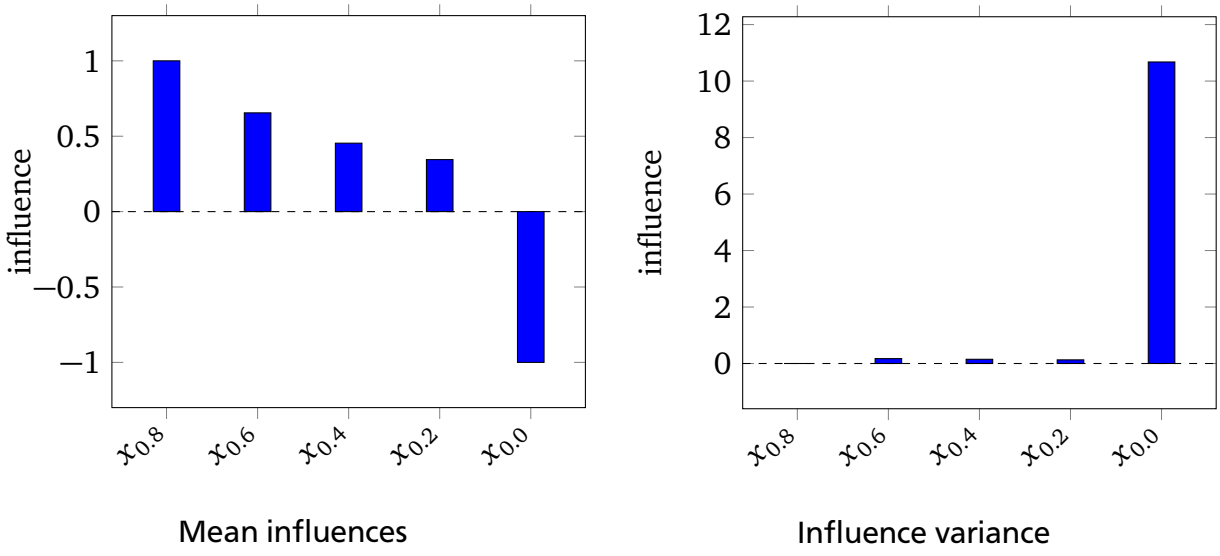


Figure 5: Mean influences and associated variances calculated by the LICON method for the complete artificial data set.

LICON: As seen in Figure 5, the LICON method detailed by Kasneci and Gottron (2016) delivers a good approximation and correctly identifies the order of the influences. Furthermore, the LICON method has a low variance for the inputs with a given correlation. For the input variable $X_{0.8}$ the variance is 0, and for the variables $x_{0.6}$, $x_{0.4}$ and $x_{0.2}$ the variance is near 0.15. However, the influence of the non-correlated input variable $x_{0.0}$ is calculated incorrectly and has a high variance of 10.676. This high variance implies that it was not possible for LICON to accurately identify the non-correlated variable. The randomness occuring in the artificial data set might be a possible explan-

ation for this fact, as even the logistic regression calculated a correlation coefficient of 0.19 (as seen in Figure 4) for the uncorrelated variable $x_{0.0}$.

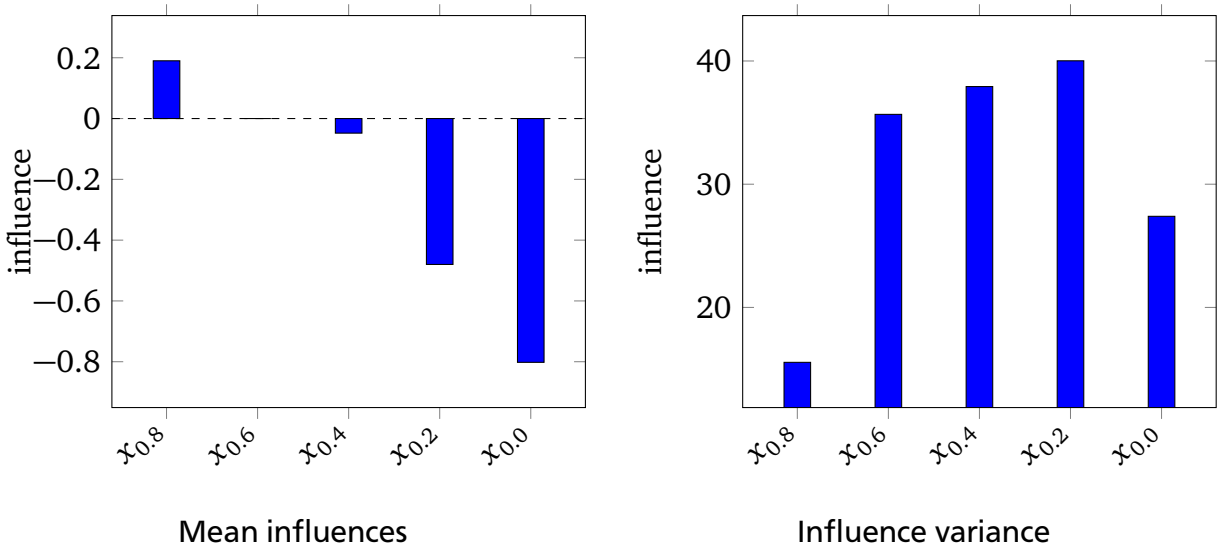


Figure 6: Mean influences and associated variances calculated by the GSA method for the complete artificial data set.

GSA: Like the LICON method, the GSA method identified the same order between the input variables. If only the positive influences had been taken into account, the method performed done well (as the calculated positive influences are $x_{0.8} = 1.0$, $x_{0.6} = 0.749$, $x_{0.4} = 0.659$, $x_{0.2} = 0.343$ and $x_{0.0} = 0.058$). However, the method also produced negative influences. Thus, the influence values differ considerably from those calculated by the LICON method and from the expected values, as seen in Figure 6. Only the $x_{0.8}$ input variable exhibits a positive influence on the output variable. For the $x_{0.6}$ variable no influence is testified, while for all other variables a negative influence is calculated. This poor performance of the GSA method is further confirmed by the relatively high variance values (around 30). Thus, the GSA method was not able to calculate the correct influence values for the artificial data set. However, this may only be an issue with this specific data set; therefore, the results of the other two data sets have to be examined.

PLAY: Finally, Figure 7 shows that the PLAY method calculated influences for the artificial data set more accurately than the GSA method, but less so than the LICON method. The PLAY method produced the same mistake as the LICON method concerning the influence of the uncorrelated variable x_0. This is understandable because the PLAY method is based on LICON. The PLAY method was able to return the correct order of

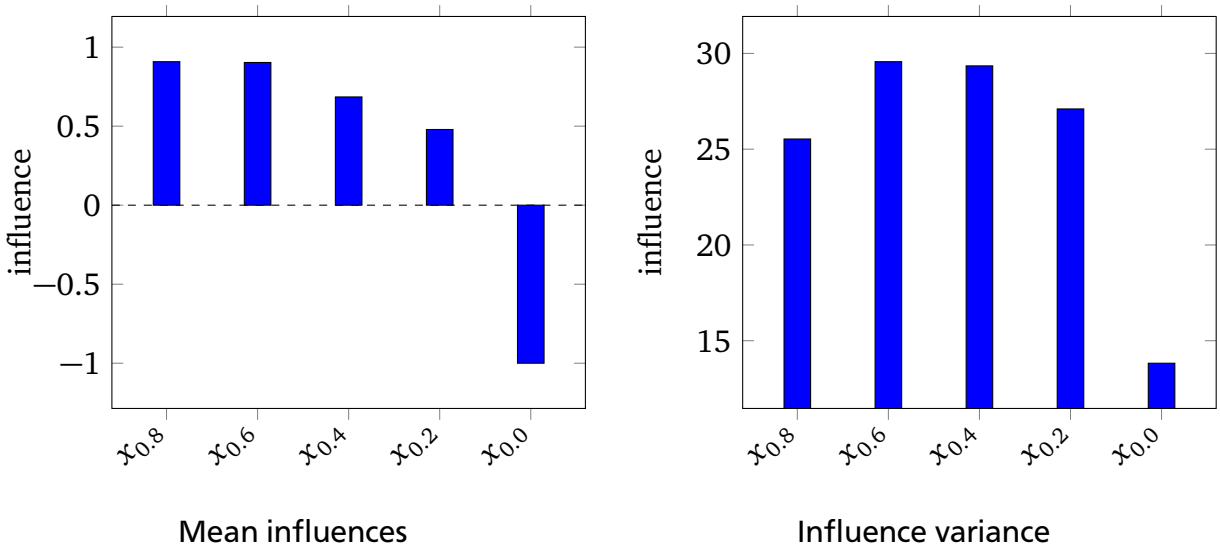


Figure 7: Mean influences and associated variances calculated by the PLAY method for the complete artificial data set.

influence values ($x_{0.8} = 0.908$, $x_{0.6} = 0.903$, $x_{0.4} = 0.685$, $x_{0.2} = 0.479$ and $x_{0.0} = -1$). However, it overestimated the influence of the input variables $x_{0.6}$, $x_{0.4}$ and $x_{0.2}$. Due to the relatively high variances (around 25) the accuracy of the calculated influence values could thus be questioned.

Evaluation of method selectivity

To compare the selectivity of the chosen methods, the AUC and GINI metrics of a logistic regression model (trained by using a 10-fold cross validation on the 'influence data set' for each method) were evaluated.

The resulting ROC curves and model adjustment curves can be seen in appendix J, 'ROC Curves for the Logistic Regression Models on the Influence Data Sets', and appendix K, 'Model Adjustments for the Logistic Regression Models on the Influence Data Sets'. For the LICON method it took 1,000 training iterations were required on the influence data set to reach the state where the model did not improve further. In contrast, the logistic model trained on the influence data set created by the PLAY method only required 300 iterations to reach that state, while the logistic model for the GSA method achieved the 'final' state after 100 iterations. As a fast model adjustment indicates a data set with a parameterisation for the model that is easy to find, the GSA method could be

seen as superior to the PLAY method, which in turn can be considered superior to the LICON method.

Measure	Target class	Non target class	Weighted mean
AUC	0.6808472835932691	0.6808472835932693	0.6808472835932691
GINI	0.3616945671865381	0.3616945671865383	0.36169456718653814

Table 5: Measure results for the logistic regression model trained on the influence data set generated by the LICON method for the artificial data set.

Table 5 presents the AUC and GINI results for the logistic regression model trained on the influence data set generated by the LICON method. It indicates that the LICON method achieves a good AUC value of ~ 0.68 and an acceptable GINI value of ~ 0.36. Therefore, it has an acceptable selectivity.

Measure	Target class	Non target class	Weighted mean
AUC	0.7467813777829202	0.7467813777829202	0.7467813777829202
GINI	0.4935627555658405	0.4935627555658405	0.4935627555658405

Table 6: Measure results for the logistic regression model trained on the influence data set generated by the GSA method for the artificial data set.

The AUC and GINI values of the GSA method, shown in Table 6, are substantially better than those calculated by the LICON method. Hence, as the logistic regression model based on the influence data set generated by the GSA method is more selective than that produced by the LICON method, this indicates that the GSA method is more selective than the LICON method.

Measure	Target class	Non target class	Weighted mean
AUC	0.7369441274456914	0.7369441274456915	0.7369441274456914
GINI	0.47928825489138294	0.47928825489138294	0.47928825489138294

Table 7: Measure results for the logistic regression model trained on the influence data set generated by the PLAY method for the artificial data set.

For the PLAY method, the resulting AUC and GINI values of the logistic regression model trained on the influence data set can be seen in Table 7. Notably, the AUC value of ~ 0.737 and the GINI value of ~ 0.479 are very close to the values of the GSA method, and, therefore, are undoubtedly better than the values of the LICON method. This indicates that the PLAY method is more selective than the LICON method for the artificial

data set. Hence, the combination of LICON with the profile approach of the GSA method seems, at least, useful for selectivity concerning the artificial data set.

5.1.4 Assessment of the Explanation Methods Regarding Artificial Data

For the artificial data set, only the LICON method is able to calculate influence values near to the true correlation between the input variables and the output variables. Although the LICON approach is effective at determining the 'true' influences, it achieves below-average values regarding the selectivity measures for complex models (Schulte-Mattler, Daun, and Manns, 2004).

The GSA method, on the other hand, attains acceptable values for selectivity measures but completely fails to extract the correct influence values. One reason for this inability to calculate the correct influences might be the gradient sensitivity measure used in the GSA method, which works on the network's output. This might also explain the optimal selectivity values of the three methods. The DANN model, for which the influence values were calculated, achieved an AUC value of ~ 0.743 and a GINI value of ~ 0.486. Therefore, as the GSA method is largely based on the predicted outputs of the neural network, it might be reasonable to believe that the good selectivity values of the GSA method are partly due to the good values of the examined DANN model.

The PLAY method combines the ability to (nearly) identify the correct influence values of the LICON method with the desirable selectivity of the GSA method. Therefore, the PLAY method seems to be an enhancement of the LICON method or, in other words, a good sensitivity measure for the GSA method. In summary, it can be stated that although the PLAY method would benefit from some further tweaking, it has proven to be valid and comparable to existing methods, at least for the artificial data set. Thus, for the artificial data set the null hypothesis can be rejected.

5.2 German Credit Data Set

5.2.1 Data Set

The second step in the evaluation of the methods was the test on ‘real-world’ data. The German credit data set produced by Hofmann (cited in Dua and Karra Taniskidou, 2017) was chosen because it imitates the data used in the financial world; this is a field that demands reliable explanation methods for DANNs in order to utilise them in the future. The data set consisted of 20 input variables and one binary output variable. The input variables included continuous and discrete variables which cover features like the ‘credit duration’ and the ‘amount’, the ‘credit history’, the ‘purpose’ and the ‘age’, for example. All of the features were used to classify a consumer as a ‘good’ or a ‘bad’ consumer concerning the credit risk.

For application as a training data set in the experiment, all numerical variables of the German credit data set were transformed to an interval between 0 and 1. Furthermore, the discrete variables were one-hot encoded into binary vectors. To evaluate the influence values calculated by the examined methods, the correlation coefficients of a logistic regression model were taken as a benchmark. This was done because the logistic regression model is an accepted and implemented statistical model in the financial industry (Ploeg, 2010).

The logistic regression model was trained using a 10-fold cross validation implemented by ‘Weka 3 Data Mining Software’ (Frank, Hall, and Witten, 2016). As the logistic regression was applied to the German credit data set, it retrieved the correlation coefficients seen in Figure 8. It is reasonable that the ‘duration’, ‘credit history’ and ‘credit amount’ are presented as the three major variables that negatively affect the rating of a consumer, as all three features clearly increase the risk of a credit-granting institution. These three variables are followed by three variables with a notable but minor negative effect: ‘instalment commitment’, ‘personal status’ and ‘existing credits’. This is also intuitively comprehensible: a higher instalment commitment reduces the disposable income of the debtor, and a higher chance of a cancelled payment results as the debtor has less liquid money for unforeseen events. This risk is intensified by other existing credits

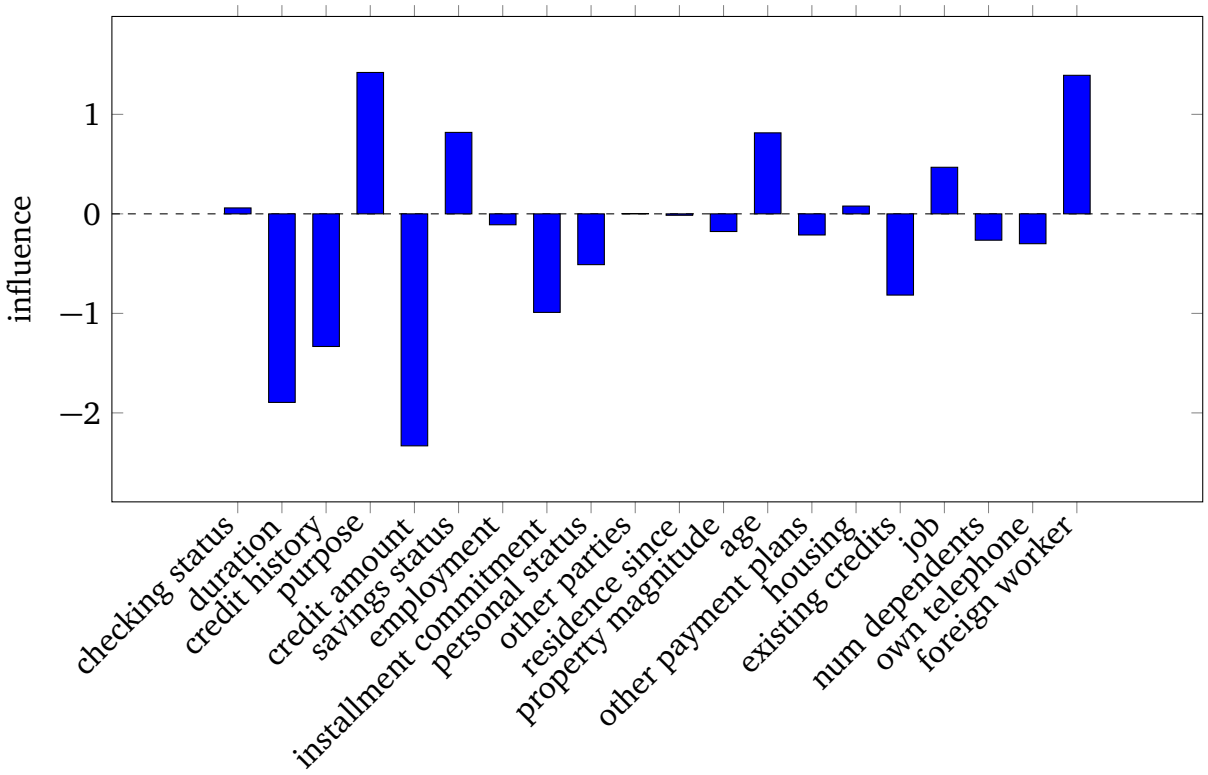


Figure 8: Logistic regression correlation coefficients for the German credit data set.

and the personal status of a debtor (single, married or divorced). The positive correlation coefficient values of the variables ‘purpose’, ‘saving status’, ‘age’, ‘job’ and ‘foreign worker’ intuitively make sense for the variables ‘saving status’, ‘age’ and ‘job’. However, the variable ‘purpose’ could be expected to be either a positive or a negative influence, depending on the purpose of the credit. This is the case if the correlation coefficients of the individual manifestations are examined. The result for the variable ‘foreign worker’ is understandable when it is considered that the calculated correlation coefficient is associated with the manifestation of ‘no foreign worker’. Therefore, it is reasonable that the logistic regression rates not being a foreign worker as a positive influence.

5.2.2 Neural Network Architecture

For the evaluation of the methods using the German credit data set, a neural network was created. The architecture of the network contains one input layer with 74 inputs, two hidden layers and one output layer. In the first hidden layer, 33 neurons were applied which used the logistic function as the activation function. For the second hidden layer, 23 neurons were implemented which used a rectifier as the activation function.

The output layer was from out of two neurons which used the softmax function for activation.

After 310 iterations on the full German credit data set the training was stopped because the model did not improve further, as seen in appendix F, 'Model Adjustments for the DANN Models Trained on the Artificial and German Credit Data Sets'. The resulting ROC curves can be seen in appendix C, 'ROC Curves for the German Credit Data Set'.

Measure	Target class	Non target class	Weighted mean
AUC	0.7348991582200977	0.7348191443730302	0.7348917400461129
GINI	0.4697983164401953	0.4696382887460607	0.4697834800922256

Table 8: Measured results for the 10-fold cross validation training of the DANN used on the German credit data set.

Measure	Target class	Non target class
Precision	0.8769230769230769	0.6
Recall	0.8028169014084507	0.7241379310344828
F-Measure	0.8382352941176471	0.65625
Accuracy	0.78	0.78
AUC	0.805245264691598	0.8052452646915979
GINI	0.6104905293831959	0.6104905293831957

Table 9: Measured results for the selected DANN and a decision threshold of 0.5 model used on the German credit data set.

Table 8 indicates the achieved AUC and GINI values following the 10-fold cross validation training. The resulting weighted mean AUC value of ~ 0.734 and the GINI value of ~ 0.469 imply that the used network architecture is capable of successfully predicting classes based on the artificial data set. Furthermore, the AUC value over all model instances of the 10-fold cross validation (~ 0.734) is similar to that of the logistic regression model (~ 0.785), whose correlation coefficients are assigned as the expected influence values. Additionally, the GINI value of ~ 0.469 corroborates an expected degree of selectivity (Schulte-Mattler, Daun, and Manns, 2004).

The model with the highest AUC, GINI, accuracy and f-measure was selected for the experiment. As seen in Table 9, the selected model achieved a precision of ~ 0.877, a recall of ~ 0.803, an f-measure of ~ 0.838, an accuracy of ~ 0.78 and an AUC value of ~ 0.805 for the target class, while producing slightly less desirable values for the non-target class. For comparison, the logistic regression model used to calculate the expected

correlations between the input variables and the output variable achieved a precision of ~ 0.798, a recall of ~ 0.864, an f-measure of ~ 0.83, an accuracy of ~ 0.752 and an AUC value of ~ 0.785 for the target class. The neural network model is therefore equal to or slightly better than the logistic regression model. Additionally, the achieved GINI value of ~ 0.61 is comparable to accepted GINI values in the rating industry (ibid.). Therefore, it has been demonstrated that the trained DANN model is eligible for use with the German credit data set.

5.2.3 Experimental Results

Comparison of mean influence values and variance

The experiment using the German credit data set was performed with the aim to verify the ability of the three selected methods to calculate the influences of an input variable on the output variable for 'real-world data'. The correlation coefficients calculated by the linear regression model are used as the expected influence values because the linear regression model is an accepted tool in the domain of the data (the financial domain). Therefore, a method was considered to work accurately if it returns the correlation values of the linear regression model.

LICON: As seen in Figure 9, the LICON method produced by Kasneci and Gottron (2016) correctly identified the variables 'duration', 'credit amount', 'instalment commitment', 'personal status' and 'existing credits' as negative influences. Furthermore, it correctly identified the variables 'saving status', 'job' and 'foreign worker' as positive influences. However, the LICON method rated the variables 'credit history', 'other payment plans' and 'own telephone' as a positive influence, while the linear regression model designated them as a negative influence. In contrast, the variable 'age' was assigned a minor negative influence while the linear regression model showed otherwise. Additionally, the variable 'purpose', which has a strong positive influence in the logistic regression model, was given an influence close to zero.

However, the differences between the results of the LICON method and the logistic regression seem reasonable. For example, the variable 'credit history' did not only contain unpaid or delayed credit rates but also credits that were paid on time. Hence, it makes sense that a consumer who does pay their credit rates on time is considered a low

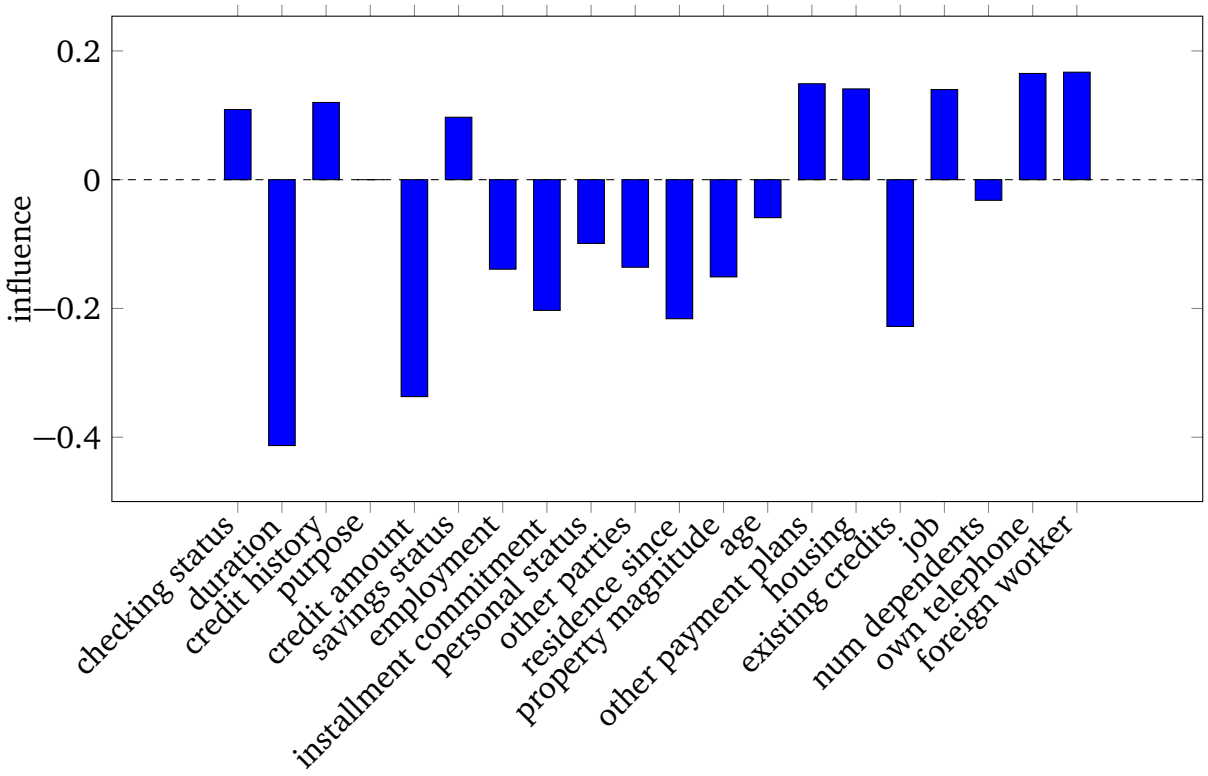


Figure 9: Mean influences as calculated by the LICON method for the complete German credit data set.

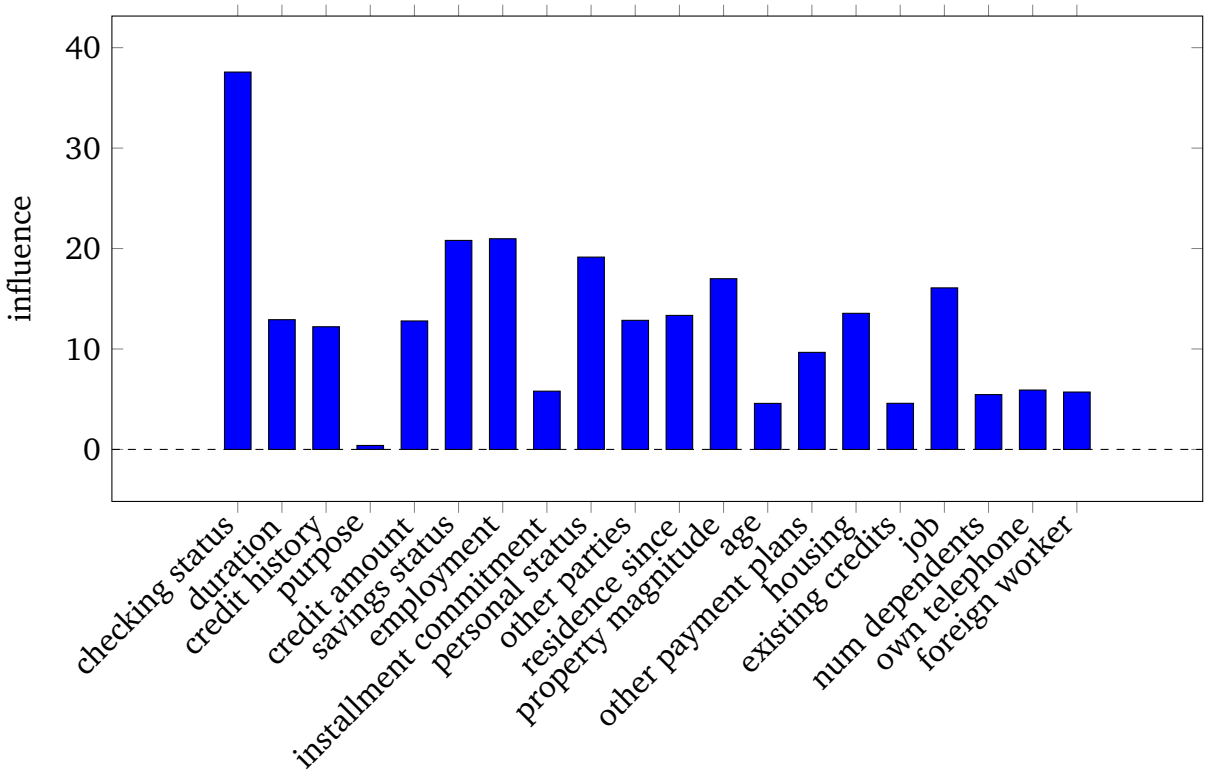


Figure 10: Variance of the influences as calculated by the LICON method for the complete German credit data set.

risk and therefore a good consumer. Furthermore, if a customer has payment plans at other banks, it seems reasonable to expect a basic trust in the reliability of the customer because other banks trust the customer. The near zero influence value of the variable 'purpose' mimics the intuitive expectation of this influence for this variable. Intuitively, the direction and magnitude of the influence for the variable 'purpose' would be expected to depend heavily on the value of the variable. For example, a large credit for a house should be considered a high-risk credit transaction, while a small consumer credit could be viewed as a low-risk credit transaction. Thus, the variable 'purpose' would be expected to receive positive and negative influence values. The aggregation of the positive and negative influence values should, therefore, result in an influence value near to zero. However, this assumption only holds true if the German credit data set is not biased in any direction regarding the variable 'purpose'. In fact, the assumed positive and negative influences were the case. The variable 'purpose' produced a positive mean influence value of 1 and a negative mean influence value of -1. This reflects a strong individual influence, such as the correlation coefficients of the logistic regression, paired with the expected dependency of the influence direction (positive or negative).

Finally, the relatively high variance values (as seen in Figure 10), especially for the variable 'checking status', indicate deviations in the magnitude of the influence values for the individual samples.

GSA: As seen in Figure 11, the GSA method performed substantially more poorly compared to the LICON method. It also performed more poorly for the German credit data set compared to its performance on the artificial data set. Only the variables 'checking status' and 'foreign worker' are correctly identified to have a positive influence on the output. Meanwhile, the variables 'duration', 'credit history', 'credit amount', 'instalment commitment', 'personal status' and 'existing credits' are identified as a negative influence. However, the reliability of this result can be doubted, as all variables with the exception of 'checking status' and 'foreign worker' were rated as a negative influence. Even those variables which would be expected to have a positive influence, like 'savings status' or 'job', were rated as negative. Additionally, the high variance values of between 100 and 200 further strengthen distrust concerning the GSA method as an explanation method for a DANN. This means that the sensitivity measure based on the network's

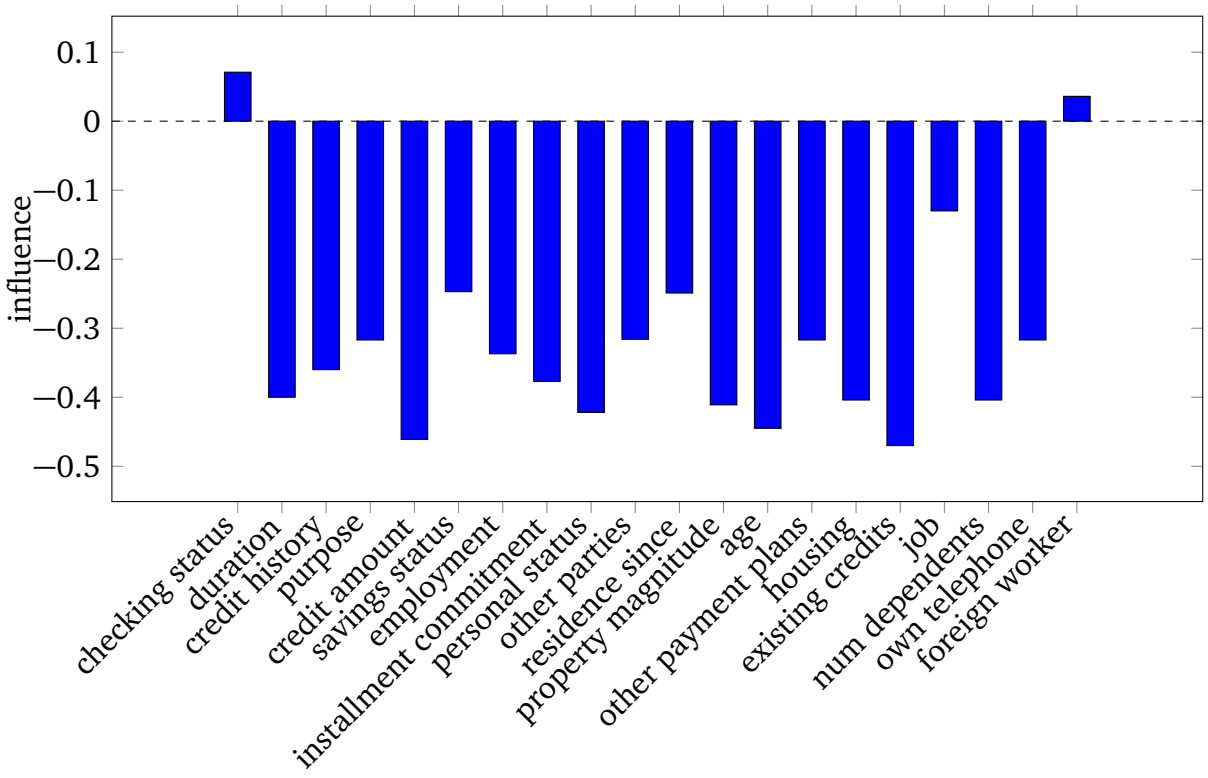


Figure 11: Mean influences as calculated by the GSA method for the complete German credit data set.

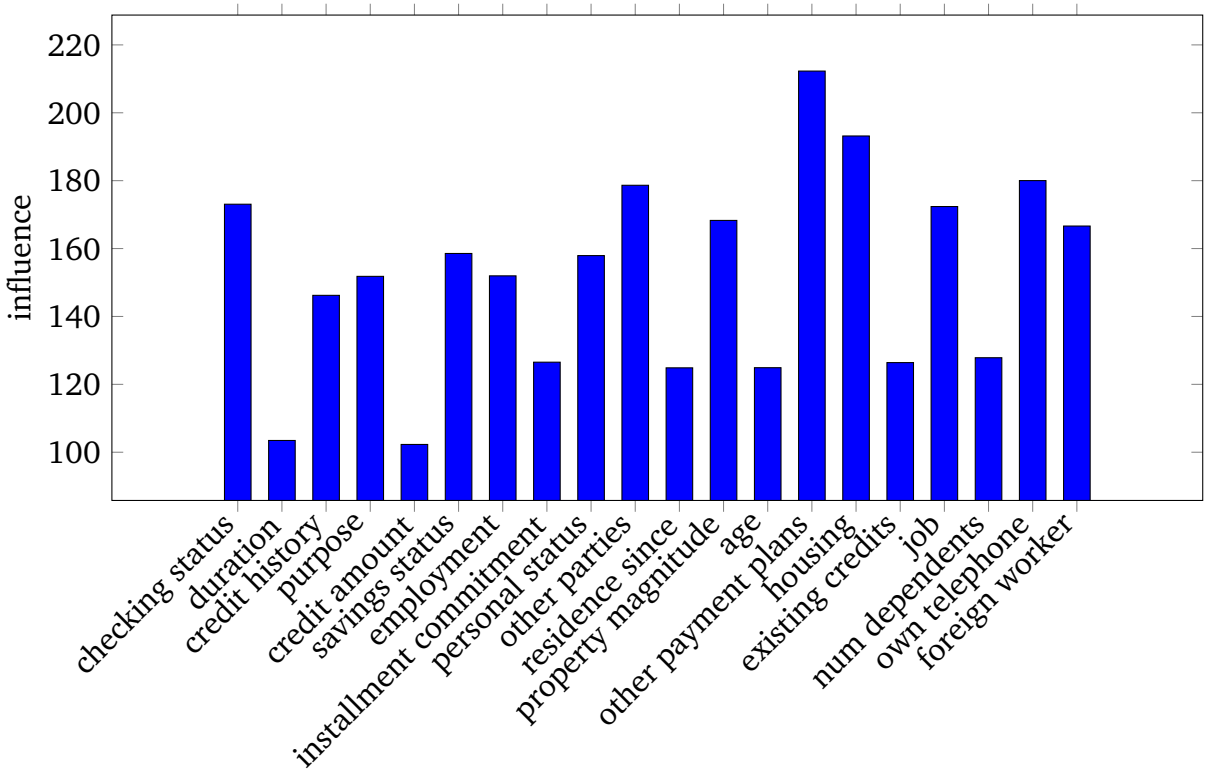


Figure 12: Variance of the influences as calculated by the GSA method for the complete German credit data set.

output (as used here for the implemented GSA method) does not appear to be reliable for the German credit data set.

PLAY: Finally, the influences for the variables 'checking status', 'duration', 'credit history', 'credit amount', 'instalment commitment', 'housing', 'existing credits', 'job' and 'foreign worker' calculated with the PLAY method are similar to the correlation coefficients of the logistic regression. That is to say, they indicate the same direction (positive or negative) while they differ in the magnitude of the influence value. However, the relatively high variance values indicate that the influence magnitudes of the individual samples are scattered over a large range.

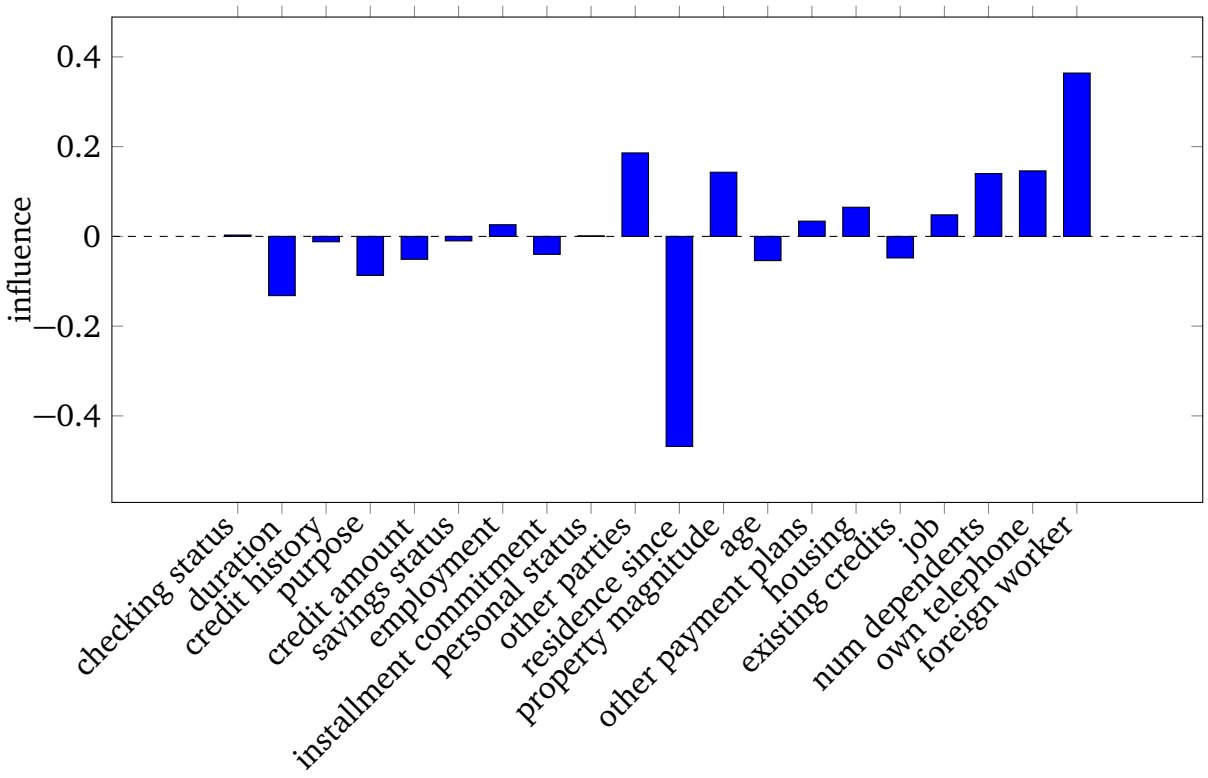


Figure 13: Mean influences as calculated by the PLAY method for the complete German credit data set.

Nonetheless, the PLAY method provides reasonable influence values. For variables with similar values to the influence values calculated by the LICON method, the same reasons mentioned in the above paragraph concerning the results of the LICON method apply here. However, the variables 'other parties', 'property magnitude' and 'num dependents', whose influence values differ from the values calculated by the LICON method and the correlation coefficients of the logistic regression model, remain intu-

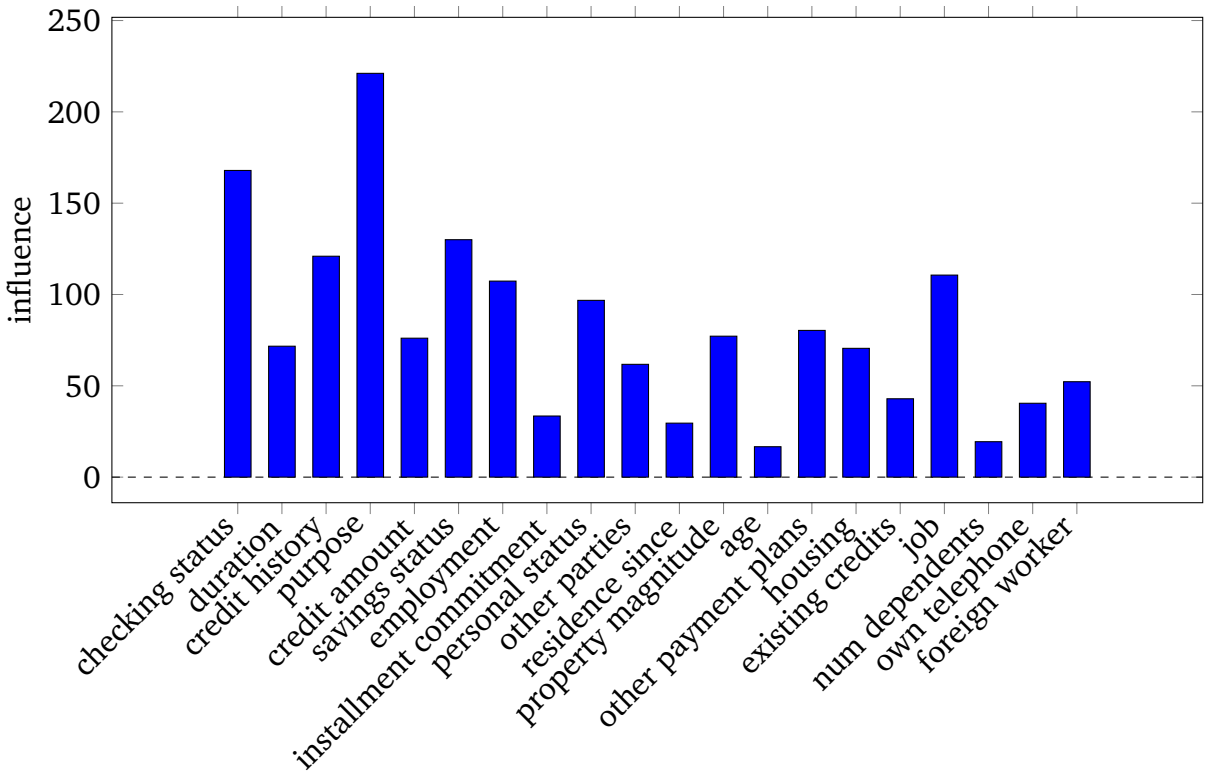


Figure 14: Variance of the influences as calculated by the PLAY method for the complete German credit data set.

itively correct. If two people apply for a credit instead of one, the risk for the bank is lowered. Furthermore, the risk is lowered if a customer can provide a property with an equivalent value as a safety or if they can rely on others being liable for payment of the credit.

Finally, it is interesting to consider that the influence value for the variable ‘purpose’ produced by the PLAY method differs from the influence value produced by the LICON method. This is despite the fact that the PLAY method is based on the LICON method. This difference, as well as the higher variances of the influence values calculated by the PLAY method, could indicate that the PLAY method is able to truly capture the relation between the input variables. One piece of evidence for this ability is the fact that the PLAY method produces different influence values for the same profile point of an input variable when the remaining variables have different values. If no connection was present between the input variables then the calculated influence would always be the same for a profile point, regardless of the value of the other input variables.

Evaluation of method selectivity

To compare the selectivity of the studied methods for the German credit data set, the AUC and GINI metrics of a logistic regression model (trained by using a 10-fold cross validation on the influence data set for each method) were evaluated.

The resulting ROC curves and model adjustment curves can be seen in appendix J, 'ROC Curves for the Logistic Regression Models on the Influence Data Sets', and appendix K, 'Model Adjustments for the Logistic Regression Models on the Influence Data Sets'. For all three logistic regression models the training was stopped after 100 iterations on the generated influence data set, as each model reached a state in which it did not improve further. Therefore, all three methods seem to have produced influence data sets which allowed a fast adjustment. This is in contrast to the artificial data set, where the adjustment speed differed significantly.

Measure	Target class	Non target class	Weighted mean
AUC	0.7342442661850359	0.7342442661850359	0.7342442661850359
GINI	0.4684885323700717	0.46848853237007165	0.46848853237007165

Table 10: Measured results for the logistic regression model trained on the influence data set generated by the LICON method for the German credit data set.

Table 10 presents the AUC and GINI results for the logistic regression model trained on the influence data set generated by the LICON method. The LICON method achieved a good AUC value of ~ 0.734 and a GINI value of ~ 0.468, both similar to the neural network for which the influences were calculated. Thus, the LICON method achieved a good selectivity overall.

Measure	Target class	Non target class	Weighted mean
AUC	0.9363181499716162	0.9363181499716163	0.9363181499716162
GINI	0.8726362999432324	0.8726362999432328	0.8726362999432324

Table 11: Measured results for the logistic regression model trained on the influence data set generated by the GSA method for the German credit data set.

The AUC and GINI values (~ 0.936 and ~ 0.872) of the GSA method, displayed in Table 11, are extraordinarily good and exceed the values calculated by the LICON method. Thus, the logistic regression model based on the influence data set generated by the GSA method is significantly more selective than that based on the influence

data set produced by the LICON method. This indicates that the GSA method is more selective than the LICON method.

Measure	Target class	Non target class	Weighted mean
AUC	0.8219526626535223	0.8219526626535224	0.8219526626535224
GINI	0.6439053253070447	0.6439053253070447	0.6439053253070447

Table 12: Measured results for the logistic regression model trained on the influence data set generated by the PLAY method for the German credit data set.

For the PLAY method, the resulting AUC and GINI values of the logistic regression model trained on the influence data set are compiled in Table 12. The AUC value (~ 0.822) and particularly the GINI value (~ 0.644) are better than the values calculated by the LICON method. Although, the PLAY method was not able to achieve excellent values for the AUC or GINI measures like the GSA method, a GINI value of ~ 0.644 is usually achieved by commercially implemented rating systems or special solution models (Schulte-Mattler, Daun, and Manns, 2004). This proves that the PLAY method is undoubtedly more selective than the LICON method for the artificial data set. Hence, the combination of LICON with the profile approach of the GSA method seems (in relation to the German credit data set) at least useful in terms of selectivity.

5.2.4 Assessment of the Explanation Methods Regarding Real-World Data

For the German credit data set, the LICON and PLAY methods were able to calculate influence values with reasonable differences to the correlation coefficients of the logistic regression. Notably, the PLAY method achieved far better values for the selectivity measure than the LICON approach. Furthermore, in the case of calculated influences of the PLAY method which differ from the results of the LICON method, these differences are understandable. One could even argue that LICON or the logistic regression calculate the influences regarding the concerned variables incorrectly. However, this is only assessed based on intuitive understanding of the variables' influence in the credit decision process. For a justified proof, the results have to be assessed by a credit risk expert.

In contrast, the GSA method achieved extraordinary values for selectivity measures but completely failed to extract the correct or even reasonable influence values. A reason for this inability to calculate the correct influences could be the gradient sensitivity

measure that is used to calculate influences. Similar to the issue in the artificial data set, the calculation of the influences based on the network's output seems unsuccessful for the German credit data set. Furthermore, this dependence on the network's output may also explain the best selectivity values of the three methods. The influence data set produced by the GSA method consisted of influences that were mainly based on the network's output. Therefore, one could argue that this influence data set consisted of nothing other than the network's outputs. So, the logistic regression, which was applied on the influence data set, attempted to classify the target class based on the outputs predicted by the DANN. The high AUC and GINI values of the GSA method could then be at least partly explained by the already high values achieved by the DANN. For all that, as the calculated influences of the GSA are far from the expected influence values the method seems to be generally inapplicable for the explanation of a DANN with the German credit data set.

The PLAY method, on the other hand, combines the ability to identify correct and reasonable influence values of the LICON method with the good selectivity of the GSA method. Therefore, as for the artificial data set, the PLAY method seems to be an enhancement of the LICON method or a good sensitivity measure for the GSA method for this data set. Thus, for the German credit data set the null hypothesis can be rejected.

5.3 MNIST Handwritten Digits

5.3.1 Data Set

In addition to the above mentioned artificial data set and the German credit data set, which tested the methods' theoretical and 'real-world' ability to calculate the correct influences, the methods were also evaluated on a benchmarking data set. This data set is the MNIST database of handwritten images, which has been used in multiple research papers to verify the accuracy of DANNs (Keysers, 2007; Mizukami et al., 2010; Simard, Steinkraus, and Platt, 2003). Classifying handwritten digits is a task that can be easily solved by a human, as a human can intuitively understand which pixel affects the classification of an image for a particular digit. Therefore, it is assumed to be equally easy for a human to assess the calculated influences of an explanation method, if the influences are presented as an image (Kasneci and Gottron, 2016).

The images of the digits in the MNIST data set have a size of 28 by 28 pixels. Figure 15 provides an example of the various possible types of handwritten digits, and illustrates how the data set contains slightly rotated, thinner or thicker and skewed digits. Additionally, some digits are written in different ways (for example digits 4, 7 and 9). They might be open or closed at the top (digit 4), ccontain a horizontal dash in the middle (digit 7) or be written with a straight line or a slight curve at the end.

The MNIST data set consists of two parts: a training set which contains 60,000 labelled images, and a test set which contains only 10,000 labelled images. For this thesis the test set with 10,000 records of handwritten digits was used, as the number of records was sufficient for the experiment. For the training of the digit-specific binary classifiers, 10 copies of the data set were created in which the values of the class variable were mapped to either the value 'TargetClass' or the value 'NonTargetClass', depending on the target class of the binary classifier. Furthermore, the values of the input variables were transformed to a value between 0 and 1.

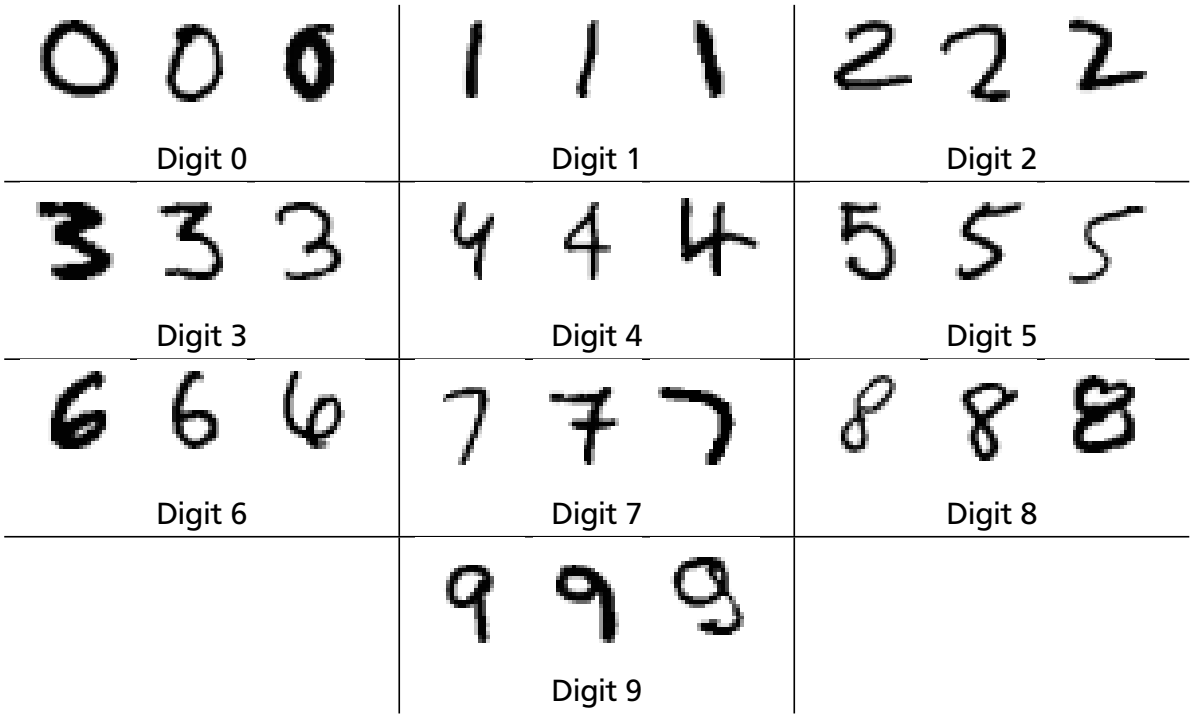


Figure 15: Examples of the entries in the MNIST database of handwritten images.

5.3.2 Neural Network Architecture

For evaluation of the methods with the MNIST data set, a neural network was created for every digit. The architecture of the networks was composed of one input layer with 784 (28×28) inputs, three hidden layers and one output layer. In the first hidden layer 33 neurons were applied, in the second hidden layer 13 neurons were implemented, and the third layer was built with two neurons. For every hidden layer the logistic function was used as the activation function. In the output layer two neurons, which use the softmax function as the activation function, were used.

After 20 iterations on the associated MNIST data set, the training of each digit-specific DANN was stopped because the model did not improve further, as shwon in appendix G, 'Model Adjustment for the DANN Models Trained on the MNIST Database of Handwritten Digits'. The resulting ROC curves of the 10-fold cross validation can be seen in appendix D, 'ROC Curves of the 10-fold Cross Validation for the MNIST Database of Handwritten Digits'. Additionally, the ROC curves of the selected model can be viewed in appendix E, 'ROC Curves of the Selected Models for the MNIST Database of Handwritten Digits'.

Measure	Target class	Non target class	Weighted mean
AUC	0.9943361476071346	0.9943361476071342	0.9943361476071342
GINI	0.9886722952142692	0.9886722952142684	0.9886722952142685

Table 13: Measured results for the 10-fold cross validation training of the DANN model used on the MNIST data set with the digit *3* as the target class.

Measure	Target class	Non target class
Precision	0.9553571428571429	0.9898648648648649
Recall	0.9224137931034483	0.994343891402715
F-Measure	0.9385964912280702	0.9920993227990971
Accuracy	0.986	0.986
AUC	0.9984006865345607	0.9984006865345609
GINI	0.9968013730691214	0.9968013730691219

Table 14: Measured results for the selected DANN model and a decision threshold of 0.5 used on the MNIST data set with the digit *3* as the target class.

Table 13 presents an example of the achieved AUC and GINI values following the 10-fold cross validation training for the network trained with the digit 3 as the target class. A full overview of the results for all 10 MNIST networks is detailed in appendix H, 'Measured results for the 10-fold cross validation training of the MNIST DANN models'. The resulting weighted mean AUC value of ~ 0.994 and GINI value of ~ 0.989 show that the used network architecture is capable of successfully predicting classes based on the MNIST data set. Additionally, the GINI value of ~ 0.989 confirms a significantly high degree of selectivity (Schulte-Mattler, Daun, and Manns, 2004).

For the experiment, the model with the highest AUC, GINI, accuracy and f-measure values was selected. A full overview of the results for all of the selected MNIST networks can be seen in appendix I, 'Measured results for the 10-fold cross validation training of the MNIST DANN models'. As seen examplarily in Table 14, the model selected for the networks that were trained with the digit 3 as the target class achieved a precision of ~ 0.955, a recall of ~ 0.922, an f-measure of ~ 0.939, an accuracy of ~ 0.986 and an AUC value of ~ 0.998 for the target class, while achieving slightly better values for the non-target class. Additionally, the achieved GINI value of ~ 0.996 indicates a high selectivity for the selected model (ibid.). Thus, the trained DANN models are eligible for use with the associated MNIST data sets.

5.3.3 Experimental Results

The experiment performed with the MNIST data set was different to those performed with the artificial and the German credit data sets, due to the fact that the MNIST data set contains 10,000 records and 784 input variables per record. The calculation of the influence values for all 784 inputs of a single record with the LICON method took around 150 milliseconds. Since the PLAY method calculates the LICON influences for six profile points for each variable, the calculation for a single record took around 11 minutes and 45 seconds. Thus, calculation of the influences for all 10,000 records of all 10 digits would take $10,000 \times 10 \times 11.76 \text{ min} = 1,176,000 \text{ min}$, which implies a time lapse of around 2.24 years. The examined data set was therefore reduced to 100 records per digit-specific data set. To obtain a well-balanced data set for the evaluation, the first 50 records of the target class and the first 50 records of the non-target class were taken. The selected records were then shuffled to prevent any bias or semi-optimal solution for the selectivity test.

Comparison of influence values for individual examples

For each of the 100 records of the digit-specific MNIST data sets, the influence values were calculated using all three methods. Figures 16 to 25 display and compare the influences of the examples given in Figure 15 for all three methods. Additionally, appendix L, 'Mean and Variance Images for the MNIST Data Sets' presents the mean and the variance over all influence values calculated for a digit-specific MNIST data set. The pixels colored in blue indicate a positive influence, while the pixels coloured in red symbolise a negative influence. The intensity of the colour refers to the magnitude of the influence.

LICON: The results calculated by the LICON method are intuitively understandable. For the digit zero, it is comprehensible that black pixels in the centre of the image (where the zero has its hallmark) would negatively influence the classification of the image as a zero. Meanwhile, the blue pixels indicating positive influence form a circle-like shape around the centre. The behaviour whereby pixels in the shape of the target

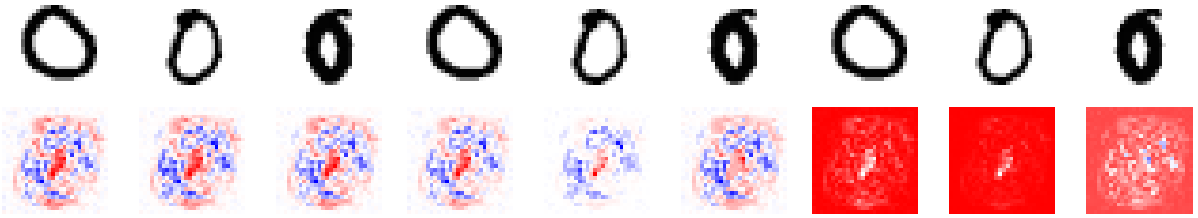
class support the class and pixels outside the shape reject it can be observed in the influence images of all digits.

Besides the direction of the influences (positive or negative), their magnitude also seems reasonable. For example, the influence images of the digits 2, 5 and 6 show strong negative influences in an area that would create a different shape or digit if the pixels were black. In the image of the digit 2, the middle area on the left-hand side is rated as a strongly negative influence. This is reasonable, as this area is usually painted black for digits like 3 or 8. For the digit 5 and 6 the top right-hand area is indicated to have a strong negative influence, because a 5 become a 9 and a 6 become an 8 if the top right-hand pixels were black.

GSA: Notably, the influences calculated by the GSA method are negative for almost all digits. The GSA method indicates a negative influence for nearly every pixel, regardless of the target class. This is a substantial difference to the influences calculated by the LICON method. Most of the influence images of the GSA method fail to exhibit any of the expected digit shapes, with the exception of images for the digit 3. In these images a silhouette of the figure 3 can be identified. However, it is not possible to find an obvious explanation for these calculated influences.

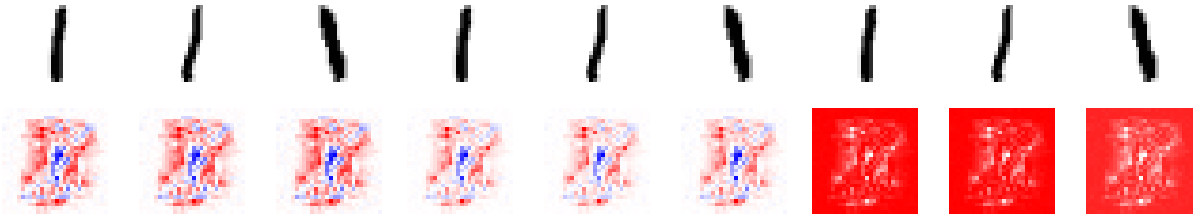
PLAY: The influences calculated with the PLAY method are similar to those calculated with LICON. A notable difference is, however, a greater distinction between the individual positive and negative influence areas. The blue and red pixels in the images created by the LICON method often join together, with only a few white pixels in between. In comparison, pixels in the images produced by the PLAY method are more concentrated and thus provide a clearer picture of the significant influence areas.

Despite these good results for the PLAY method, it should be mentioned that the influence images for the digit 7 are somewhat weak in their ability to explain the input influences. Most of the influence values in these four examples have a small magnitude and therefore seem to have no effect on the predicted output. For the digit 7, a few records produced influence values with a greater magnitude. A possible reason is that the binary classifier for the digit 7 is specialised in finding records which are in fact not 7. This seems reasonable, since the influence images produced for the digit 7 data set show stronger influences for non-target class records.



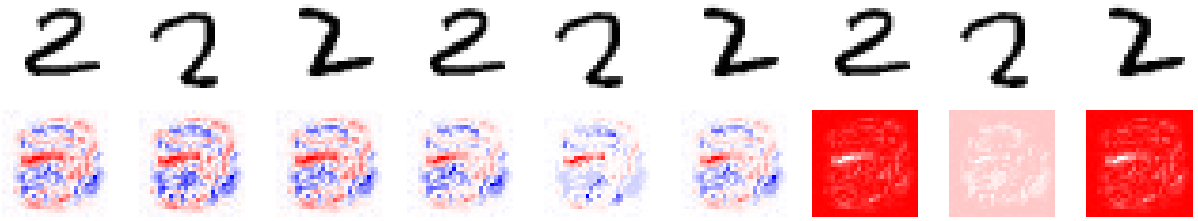
LICON PLAY GSA

Figure 16: Calculated influences for examples of the digit 0.



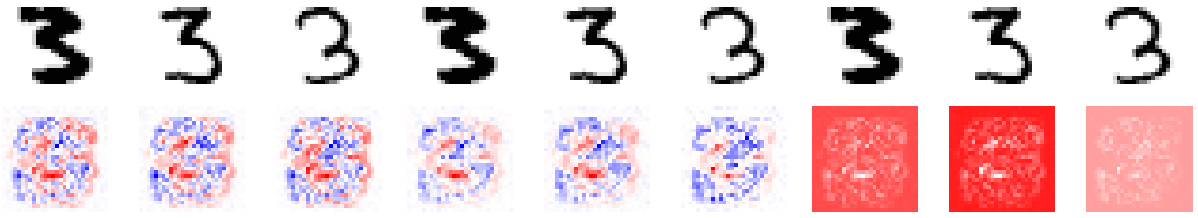
LICON PLAY GSA

Figure 17: Calculated influences for examples of the digit 1.



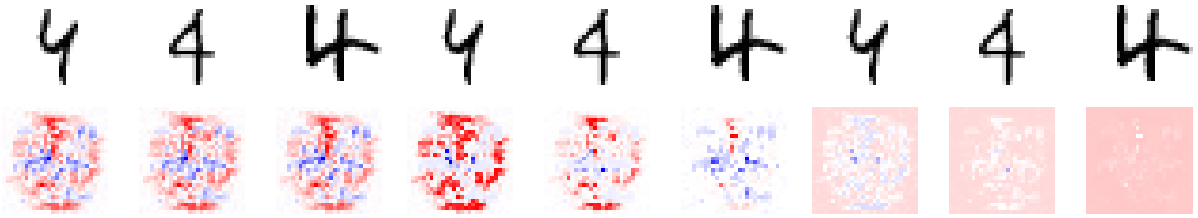
LICON PLAY GSA

Figure 18: Calculated influences for examples of the digit 2.



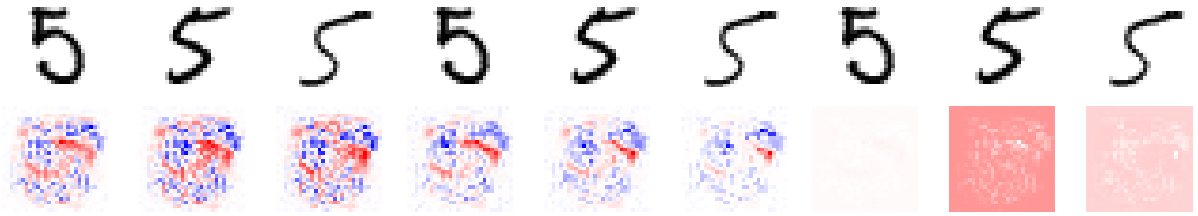
LICON PLAY GSA

Figure 19: Calculated influences for examples of the digit 3.



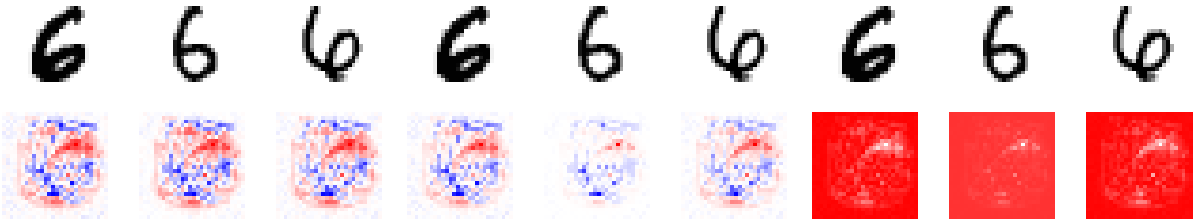
LICON PLAY GSA

Figure 20: Calculated influences for examples of the digit 4.



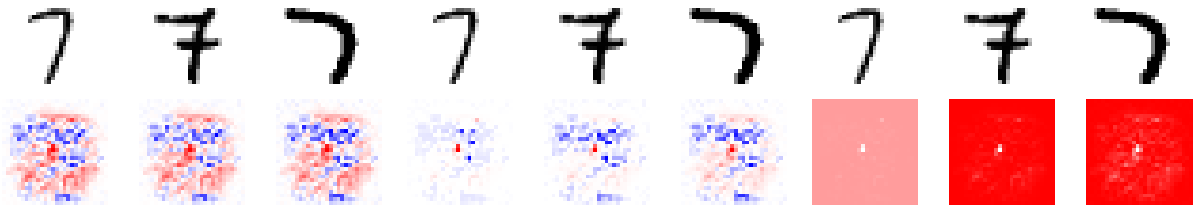
LICON PLAY GSA

Figure 21: Calculated influences for examples of the digit 5.



LICON PLAY GSA

Figure 22: Calculated influences for examples of the digit 6.



LICON PLAY GSA

Figure 23: Calculated influences for examples of the digit 7.

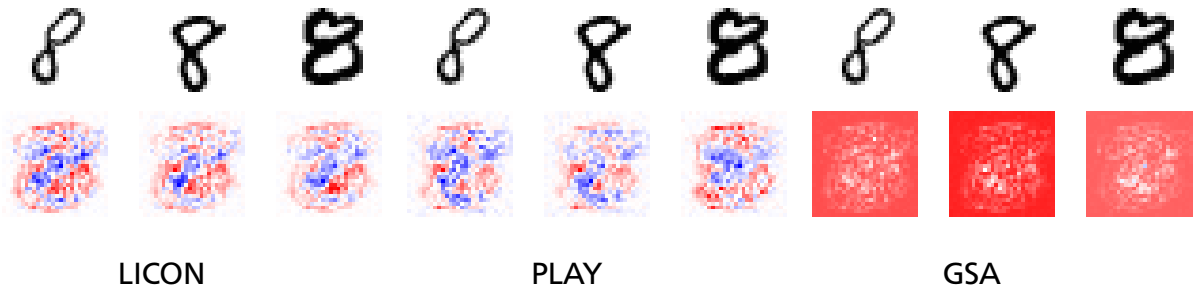


Figure 24: Calculated influences for examples of the digit 8.

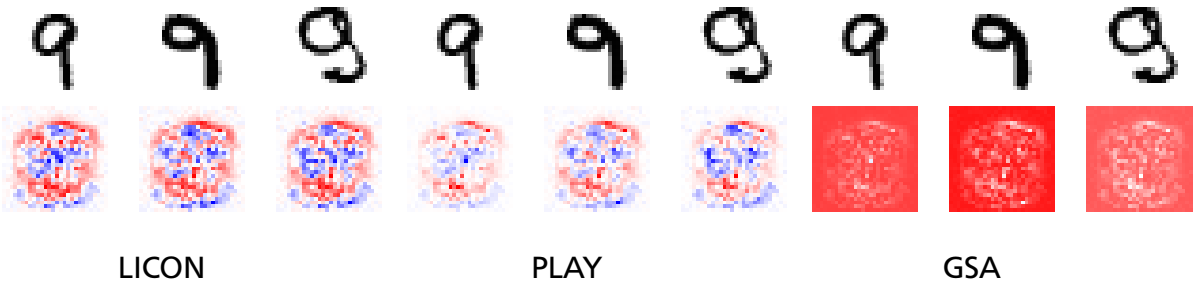


Figure 25: Calculated influences for examples of the digit 9.

Evaluation of methods selectivity

Similar to the selectivity tests performed with the influence data sets that were calculated from the artificial and the German credit data sets, a selectivity test was performed for the influence data sets produced from the digit-specific MNIST data sets. However, the results were unsatisfactory because the digit-specific MNIST data sets contained only 100 records each. A logistic regression model trained on only 100 data samples with a degree of freedom of 3,137 (1,568 positive and negative influences + 1,569 mean and variance values of the various input manifestations) is able to 'remember' the class values for every record. Thus, all of the performed logistic regressions achieved AUC and GINI values of ~ 1. As a consequence it was not possible to draw any useful or reliable conclusions from this part of the experiment. The only mentionable observation is that the logistic regression model, trained on the influence data sets constructed with the GSA method, performed slightly worse (around 0.01) than the models trained on the data sets produced by the LICON or the PLAY methods. However, the usefulness of this observation is questionable.

5.3.4 Assessment of the Explanation Methods Regarding Benchmarking Data

Even though it was not possible to perform selectivity tests for the MNIST data set due to time constraints, it is still plausible to assess the methods based on their produced influence values. The LICON method produced reasonable influence values for the MNIST data set. This is in compliance with the results of the artificial and the German credit data sets. Therefore, the LICON method once again proved to be a reliable explanation method.

In comparison, the GSA method (at least as implemented in this thesis) again failed to produce reasonable influences. The PLAY method, however, retained the capabilities of the LICON method and produced reasonable influence values. Furthermore, it improved LICON by separating or concentrating the individual influence values, resulting in a clearer view of the separate positive and negative influence areas if displayed as an image.

In summary, the prevalent hypothesis – that the PLAY method is less good and selective as the compared existing methods – can be partly rejected. It was possible to demonstrate that the PLAY method produces influence values for the MNIST data set that are comparable to or better than existing methods. However, as the selectivity was not tested for the MNIST data set, a statement concerning selectivity cannot be made.

6 Conclusions

The overall aim of this research was to explain the influence of input variables on the decision of a deep artificial neural network. The specific research objectives were:

1. To *define* the term *explanation* with respect to DANNs.
2. To *identify* existing explanation methods.
3. To *develop* a new explanation method for DANNs.
4. To *compare* the proposed method for existing explanation methods.
5. To *assess* the presented methods.

This chapter revisits the research objectives above, summarises the findings of this research work and offers the resulting conclusions.

6.1 Research Objectives: Summary of Findings and Conclusions

To accomplish the first objective, chapter 2.1 addressed the definition of explainability with regards to DANNs. The literature review examined the various uses and definitions for terms like 'explainability', 'explanation', 'interpretation', 'relevance' and 'understanding' in the context of DANNs. Since a plethora of definitions and different uses for these terms exist in connection with research that addresses explanation methods for a neural network's decisions, a lack of a commonly agreed on and consistently used definition was identified. This led to the conclusion that a research work concerned with the 'explanation' of DANNs has to clarify its aim and therefore the term itself. As a result, this thesis provided its own definition for the terms 'explanation', 'interpretation' and 'understanding' for the context of this research.

The second objective was addressed in the second part of the literature review in chapter 2.2. The conducted literature review explored the existing approaches to explain a neural network's decision described within the academic research, categorised these approaches and highlighted challenges and possibilites. It was revealed that a large number of explanation methods exist, which were classified into groups of 'connection weight', 'sensitivity analysis' and 'back propagation based' methods. Analysing

these groups of explanation methods yielded that the 'connection weight' methods were mainly developed for small networks containing a maximum number of three layers and had issues concerning their reliability. Thus, they are not applicable for the explanation of a DANN's decision. In contrast, the 'sensitivity analysis' and 'back propagation based' methods were found to be popular, applicable to DANNs and promising concerning their results. The work conducted for the second objective also included an assessment of the existing explanation methods regarding their compliance to the definitions of the terms 'explanation', 'interpretation' and 'understanding', given as a result of objective 1. This resulted in a comprehensible overview of the methods' explanation abilities.

To achieve the third objective, two existing explanation methods (GSA and LICON) from the popular method groups 'sensitivity analysis' and 'back propagation based methods' were selected. By combining the core features of the two existing methods, the new explanation method PLAY was developed. PLAY uses profiled input vectors (a feature of GSA) and a back propagation based gradient calculation (a feature of LICON) to determine the influence of an input variable on the output of a DANN.

To address objective 4 and 5, the three methods were tested on three different data sets. For the experiments, an artificial data set, a 'real-world' data set and a benchmarking data set were used. The artificial data set was generated with given correlations between the input variables and the dependent output variable in order to ensure the theoretical ability of the methods to identify the expected input influences. The 'real-world' data set was used in order to test if the influence values produced by the methods are similar to accepted statistical models. Finally, the benchmarking data set was used to test the methods' abilities on a widely used data set.

The comparison of the methods showed that the LICON method always achieved a good result for the influences, if compared to the expected values. In contrast, the GSA method produced poor influence results, though it always achieved a higher selectivity than the LICON method. The PLAY method, however, was able to combine the positive attributes of both methods. It achieved comparable (if not even better) influence results than LICON while achieving a higher selectivity. Furthermore, because of the PLAY method's construction and the additional information it returns, it takes over the good DCS values of the GSA method. Therefore, this research has shown that the new de-

veloped method PLAY is justified and the null hypothesis stated in chapter 3.1 has to be rejected. Thus the alternative hypothesis has to become the prevalent hypothesis.

6.2 Contribution to Knowledge

Questioning the contribution of this thesis to the scientific research knowledge, two aspects can be brought up: The overview of explanation methods ranked by the DCS and the uniquely proposed explanation method.

A literature review itself is not worth mentioning in this context because other researchers already used this technique to give an overview of the scientific knowledge. However, to the best knowledge of the author, so far no researcher used a score to rate the compliance of existing explanation methods to given definitions for the terms 'explanation', 'interpretation' and 'understanding'.

The proposed method PLAY is another contribution to the scientific research knowledge, even though other researchers proposed explanation methods before and used a similar experimental setup to justify their method. However, to the best knowledge of the author, no researcher before combined a profiling sensitivity analysis method with a back propagation based method. Therefore, this research offers a new approach to explain the decision of DANNs.

6.3 Limitations of the PLAY Method and Possible Future Research

Even though the PLAY method presented itself with overall good results in this research, a few points for possible improvement have to be mentioned. First of all, some of the evaluated results showed that the PLAY method is imprecise in certain cases. For example, it overestimated some of the calculated influences for the artificial data set. Therefore, further research is necessary to find potential flaws that were not detected by this research.

Another limitation of the PLAY method is its computational cost. The time taken to calculate the influence for a very complex DANN with lots of inputs shoots up by every additional evaluated profile point. Therefore, the PLAY method is currently only reasonable if applied as an in-depth examination of a classification example. For a time efficient overview of the influences for a complete data set, this research revealed that

up to date the LICON method should be preferably used. However, further research could result in an optimisation of the PLAY method's runtime and thus make the method suitable for an extended application.

Finally, this research was not able to test the influences calculated with the PLAY method for a combination of inputs and it was not it possible to test further data sets due to time constraints. Thus, the investigation of this questions may be a starting point for possible further research.

Appendices

A Overview and Use of Notation

Symbol	Explanation
M	Trained DANN model.
$\#I$	Number of inputs.
$\#L$	Number of layers in the DANN.
$\#N^l$	Number of nodes in the layer l.
$z_i^{(l)}$	The i-th Neuron in layer l.
x^0	Transformed input vector feeding into the first layer of the network.
x_i^0	i-th value of the transformed input vector, i.e. the i-th part of an input variable (e.g. discrete input variables consist of an input value for each possible state of the variable).
x^l	Output vector calculated in layer l.
x_i^l	Output value calculated for the i neuron in layer l.
w^l	Weight matrix for layer l.
w_{ji}^l	Weight connecting neuron i in layer l and neuron j in layer $l-1$. If two neurons i and j are not connected their weight is defined as 0.
b^l	Bias weights for layer l.
b_i^l	Bias weight for neuron i in layer l.
$f_i^{(l)\prime}$	Derivative of the activation function for the i-th neuron in layer l.
$\alpha_{ik}^{(l)}$	Approximated local behavior (influence) for input parameter $x_i^{(0)}$ in the linear approximation at the k-th neuron in layer l. In particular, $\alpha_{ik}^{(\#L)}$ provides the influence for input parameter $x_i^{(0)}$ on the k-th component of the output vector of the overall network.
$\beta^{(l)}$	Weighted gradients for layer l.

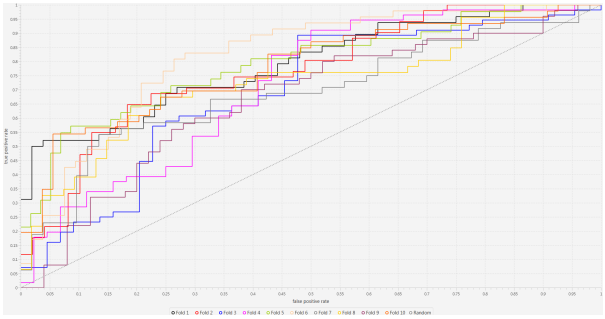
$\beta_{ij}^{(l)}$	Weighted gradient of the local linear approximation for neuron $z_i^{(l)}$ of the l-th layer and the j-th input value feeding into the neuron $z_i^{(l)}$.
X	List of input vectors x^0.
$\#X$	Number of elements in X.
$X[h]$	h-th Element of the list of input vectors X.
$X[*]$	Tail of the list. Used to add elements to the list of input vectors.
$x_b^{(0)}$	Transformed input vector used as a base for the profiling procedure.
F	Set of input variables/features to profile.
p	Number of profiles to generate for a single input variable.
$Predict(M,X[i])$	Prediction function of the model M. Predicting the output vector for the input vector $X[i]$.
$\vec{y}^{(h)}$	Output vector predicted by the model M for the input vector $X[h]$.
$\vec{y}_k^{(h)}$	The k-th element of the output vector predicted by the model M for the input vector $X[h]$. In particular, $\vec{y}_0^{(h)}$ symbolises the output for the target class and $\vec{y}_1^{(h)}$ smybolises the output for the non target class in case of a binary classifier for the input vector $X[h]$.
g_{a0}	The gradient sensitivity measure for the input variable/feature a concerning the target class. Thus, the positive influence of the input variable on the target class.
g_{a1}	The gradient sensitivity measure calculated for the input variable/feature a concerning the non target class. Thus, the negative influence of the input variable on the target class.
g	List of all gradient sensitivity measure calculated for each analysed input variable/feature in F.
$\#opt(a)$	Number of possible options for the discrete input variable/feature a.

$opt_i(a)$	The i-th option for the discrete input variable/feature a.
$max(a)$	Maximum possible value for the continuous input variable/feature a.
$min(a)$	Minimum possible value for the continuous input variable/feature a.
S_a	Sequence i.e. list of profiled values for input value/feature a.
$S_a[i]$	The i-th element of the list of profiled values for the input value/feature a.
t	Total number of input vectors created by the profiling process.
e	Number of times each entry of the profiled values list S_a has to be repeated in the $Repeat$ function for a complete combination of all profiled input variables/features.
m	Number of times each sequence/list of profiled values has to be repeated as a whole in the $Repeat$ function for a complete combination of all profiled input variables/features.
R_a	List of repeated sequences/profiled input values for the input variable/feature a.
$R_a[i]$	The i-th entry of the list of repeated sequences/profiled input values for the input variable/feature a.
$x_{ba}^{(0)}$	Part of the base vector for input variable/feature a.
$\#Y$	Number of outputs of the DANN. As the thesis is limited to a binary classifier, the number of outputs of the DANN is always 2.
A	Matrix of calculated influences ($\alpha^{(\#L)}$) for all profiled input vectors using the LICON method.
$A[h,i,k]$	The calculated influence ($\alpha_{ik}^{(\#L)}$) of input value i on the output value k for the h-th input vector $X[h]$ using the LICON method.
$\bar{A}[*,i,0]$	The mean of the calculated influences for the i-th input value of the input vector on the target class over all profiled input vectors.
m_{ik}	The mean of the calculated influences of the i-th input value on the k-th output value.

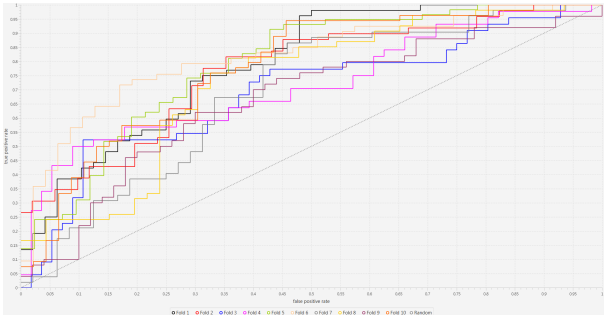
v_{ik}	The variance of the calculated influences of the i-th input value on the k-th output value.
p	The defined correlation of input variable x_p with the dependent class variable y.
x_p	An input variable with a correlation of p to the dependent class variable y.

Overview and use of notation.

B ROC Curves for the Artificial Data Set

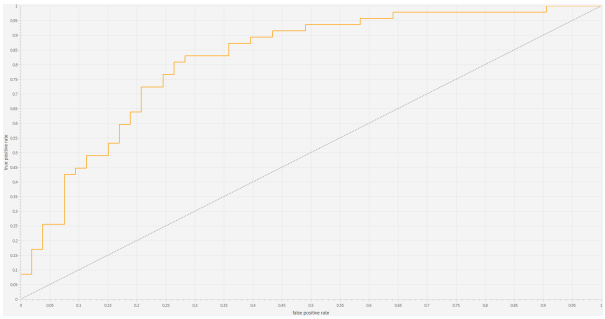


Target class

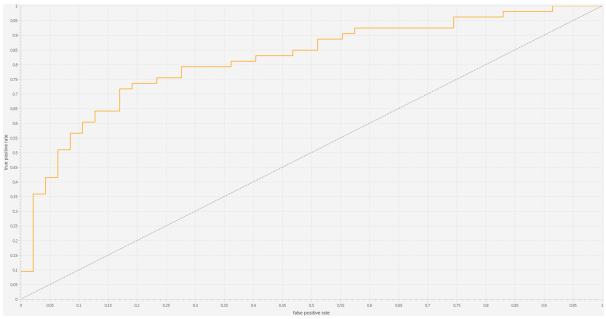


Non target class

ROC curves of every model of the 10-fold cross validation for the artificial data set.



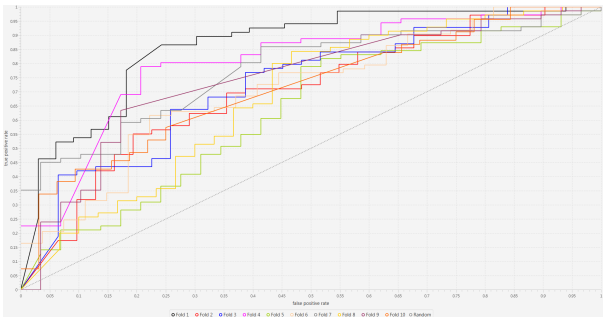
Target class



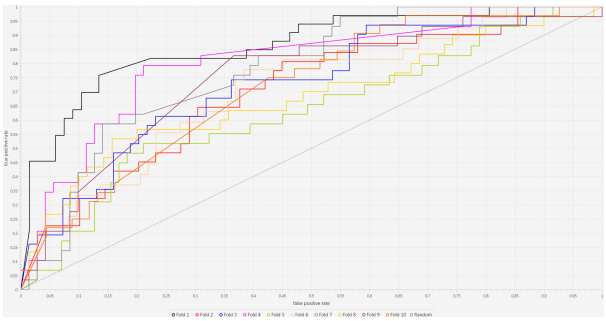
Non target class

ROC curves of the selected DANN model for the artificial data set.

C ROC Curves for the German Credit Data Set

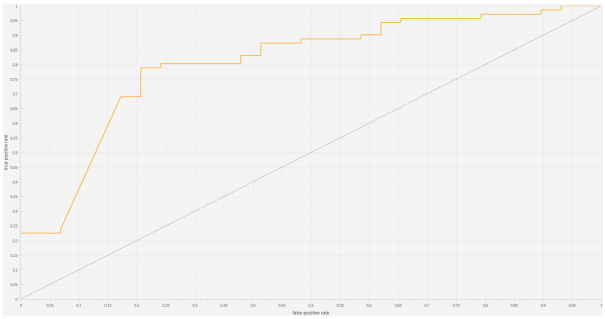


Target class

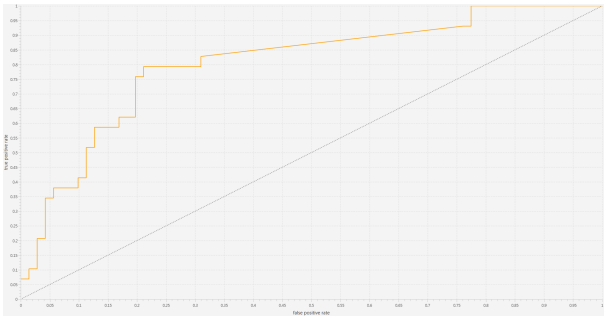


Non target class

ROC curves of every model of the 10-fold cross validation for the German credit data set.



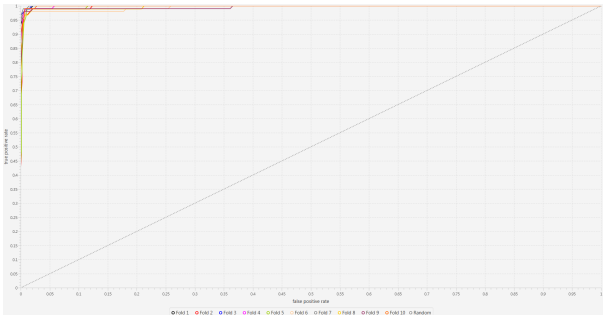
Target class



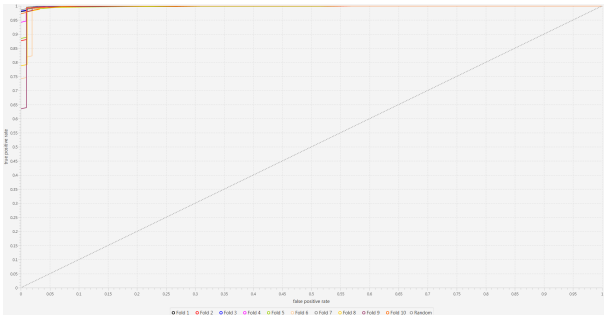
Non target class

ROC curves of the selected DANN model for the German credit data set.

D ROC Curves of the 10-fold Cross Validation for the MNIST Database of Handwritten Digits

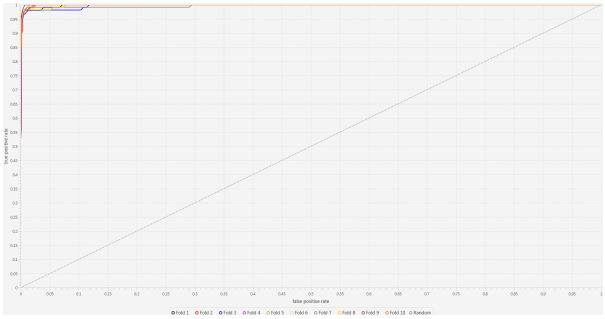


Target class

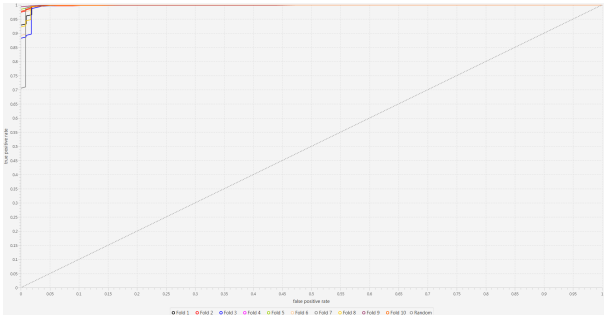


Non target class

Target class *0*

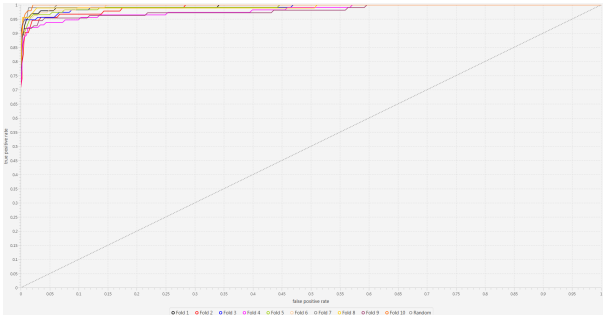


Target class

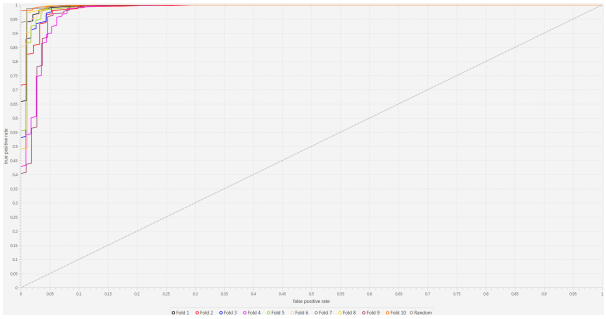


Non target class

Target class *1*

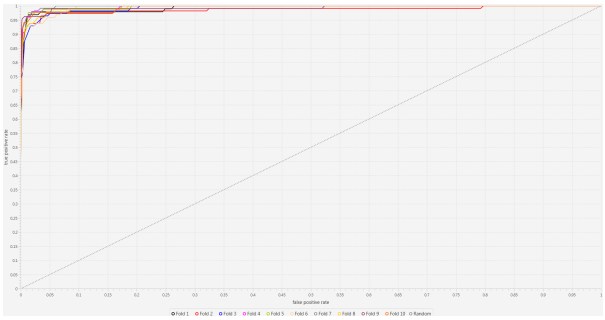


Target class

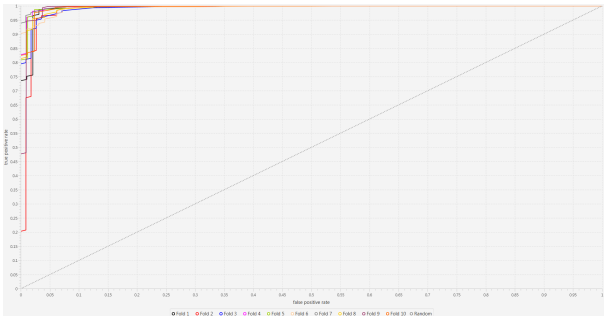


Non target class

Target class *2*

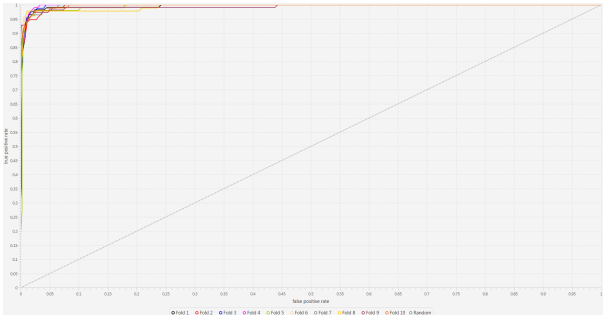


Target class

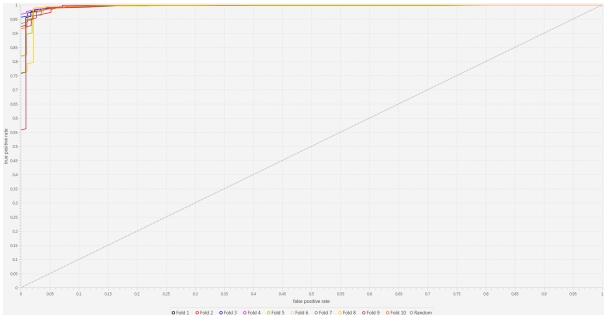


Non target class

Target class *3*

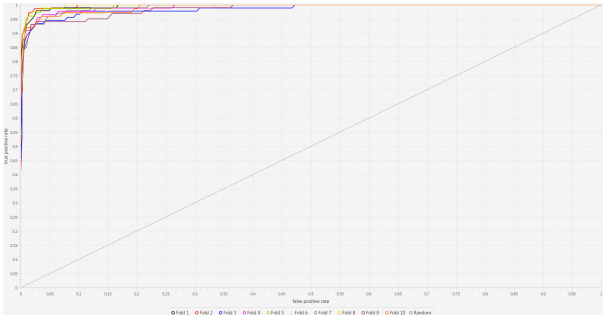


Target class

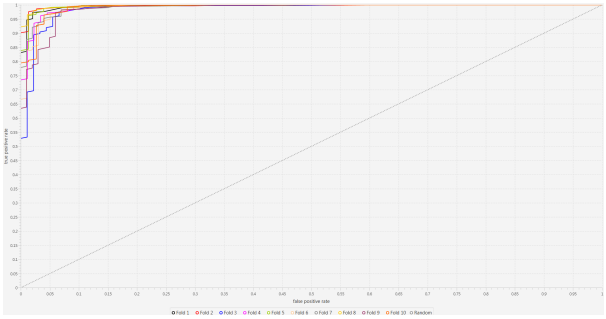


Non target class

Target class *4*

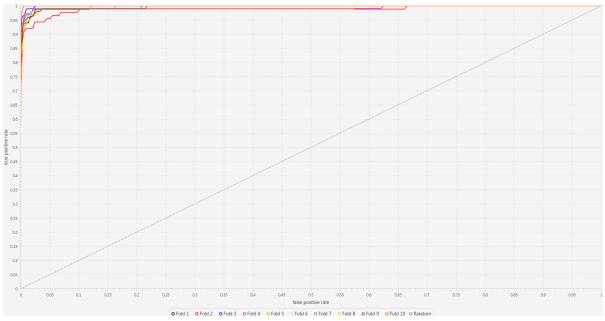


Target class

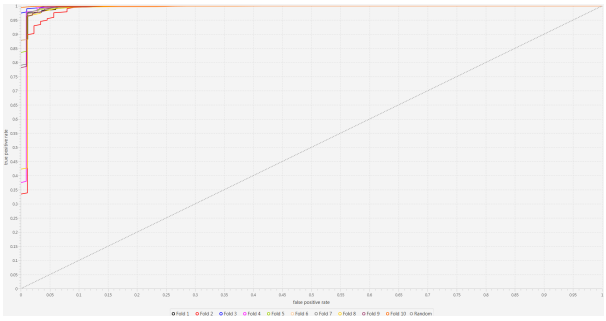


Non target class

Target class *5*

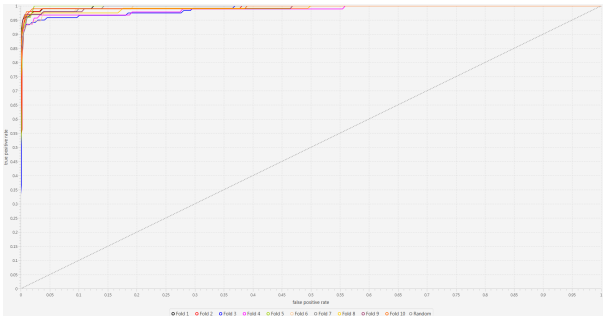


Target class

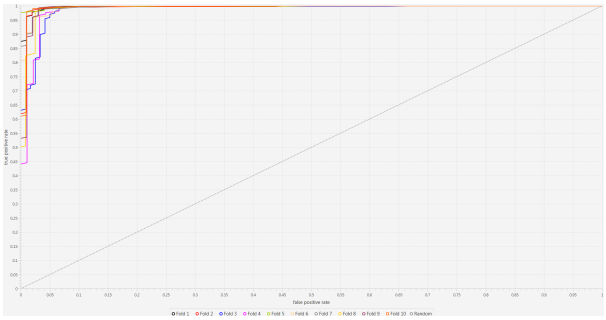


Non target class

Target class *6*

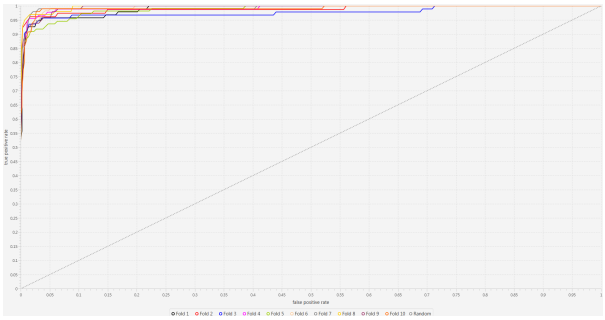


Target class

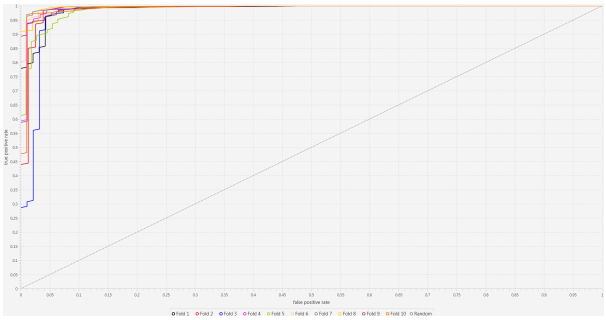


Non target class

Target class *7*

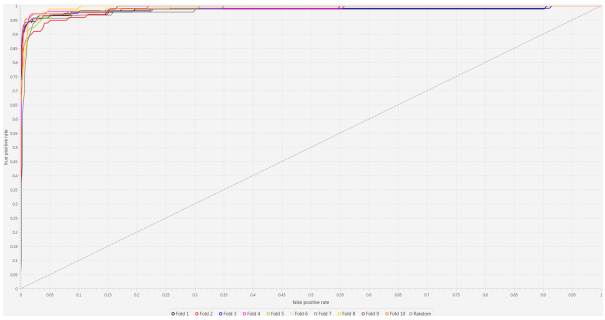


Target class

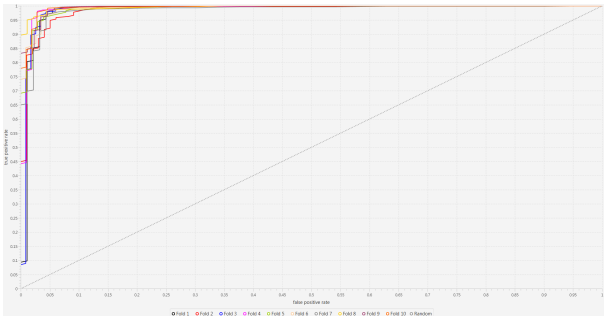


Non target class

Target class *8*



Target class

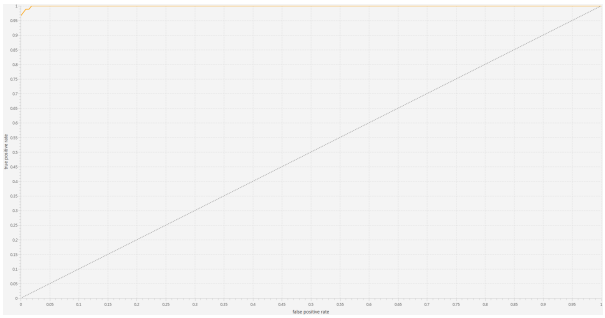


Non target class

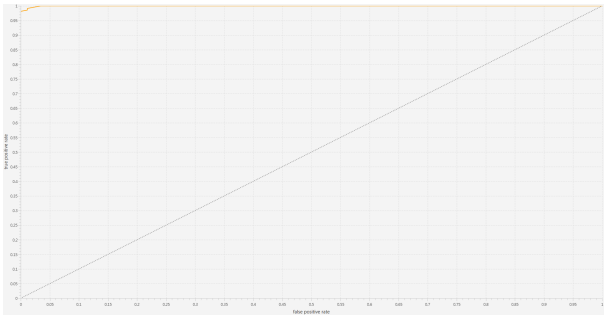
Target class *9*

ROC curves of every model of the 10-fold cross validation for the MNIST data set.

E ROC Curves of the Selected Models for the MNIST Database of Handwritten Digits

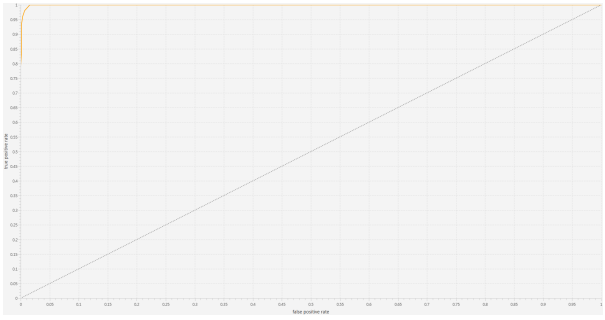


Target class

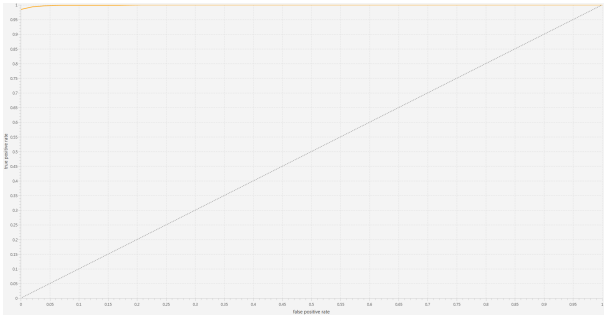


Non target class

Target class *0*

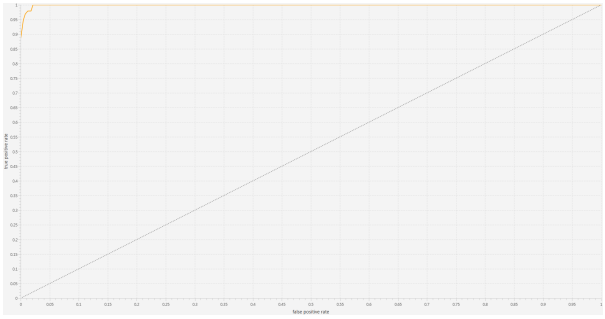


Target class

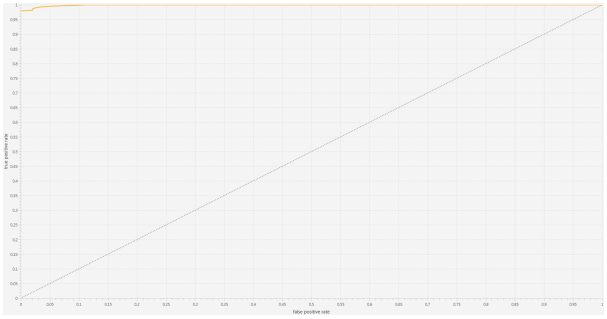


Non target class

Target class *1*

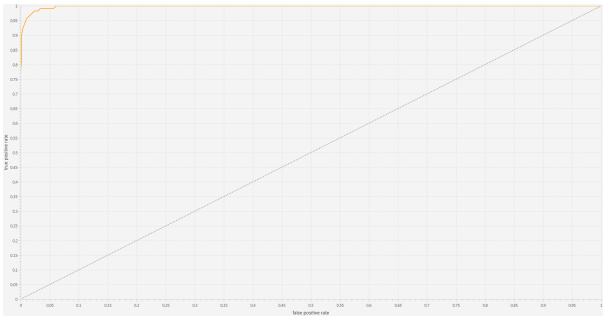


Target class

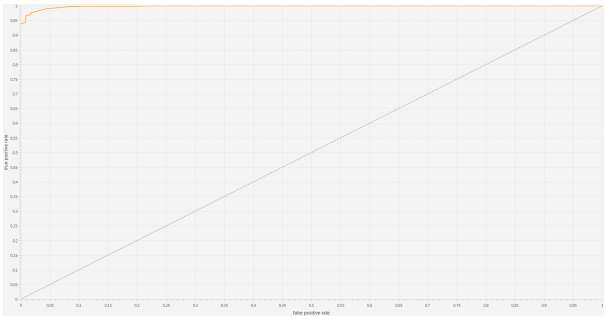


Non target class

Target class *2*

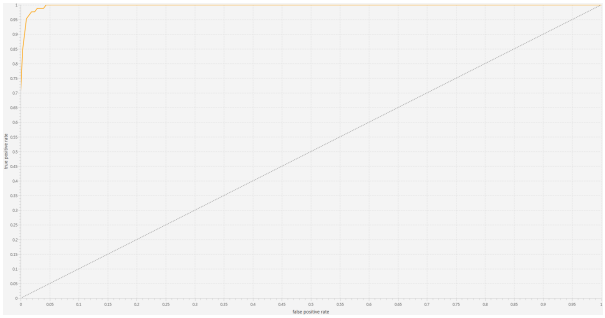


Target class

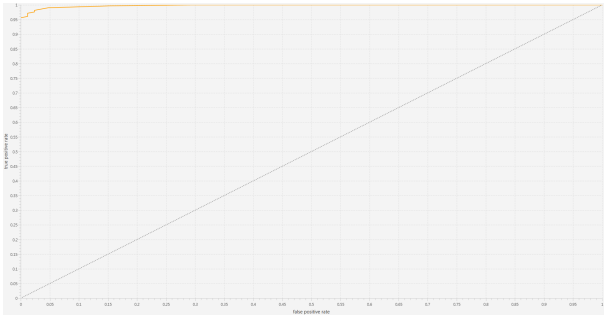


Non target class

Target class *3*

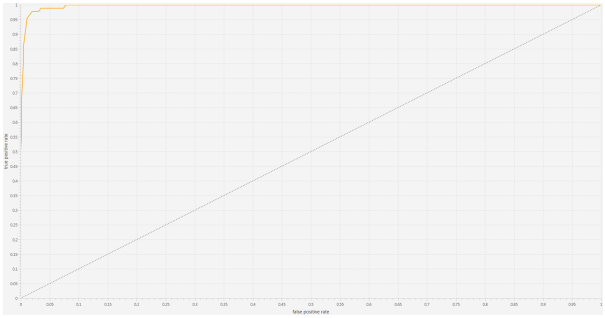


Target class

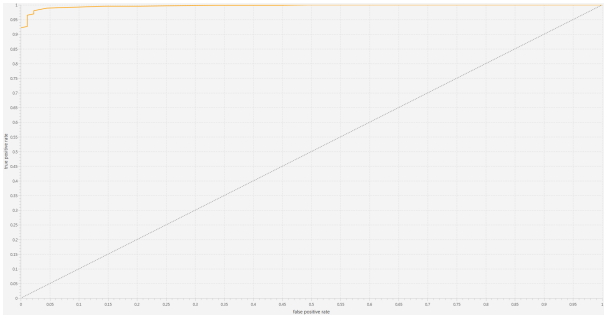


Non target class

Target class *4*

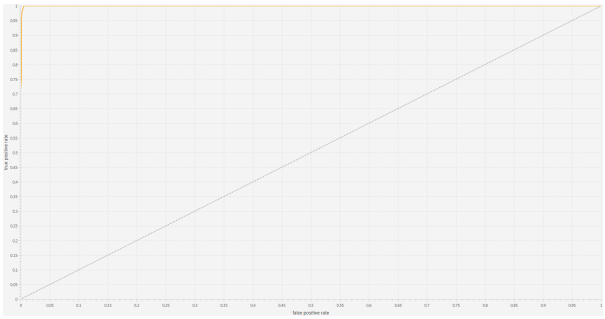


Target class

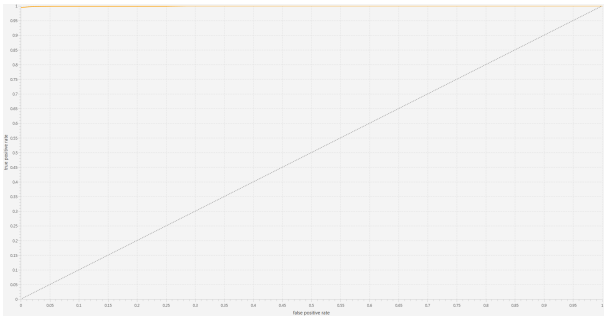


Non target class

Target class *5*

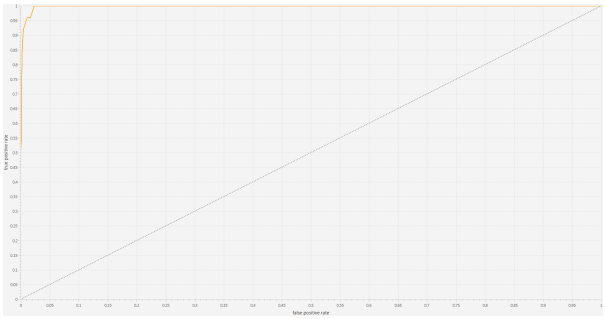


Target class

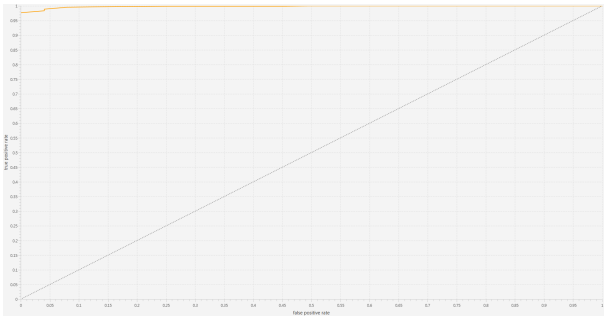


Non target class

Target class *6*

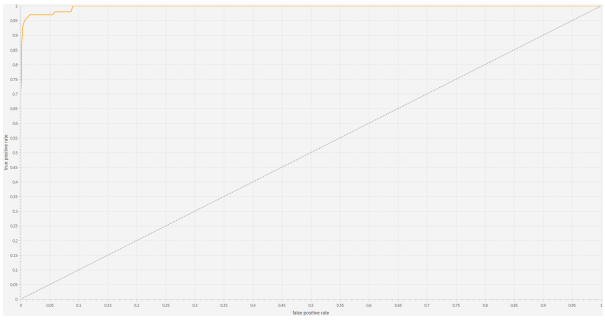


Target class

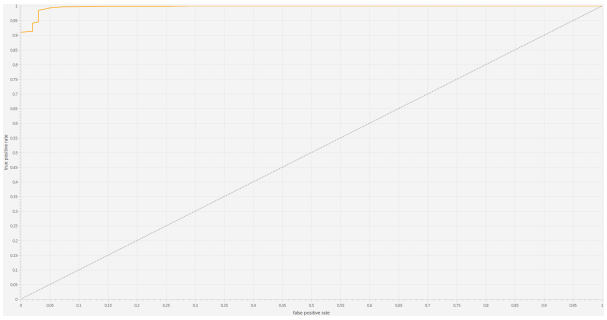


Non target class

Target class *7*

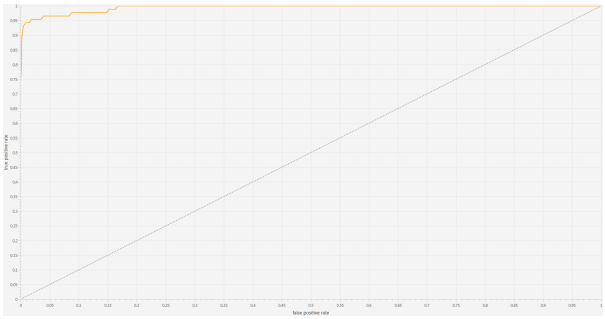


Target class

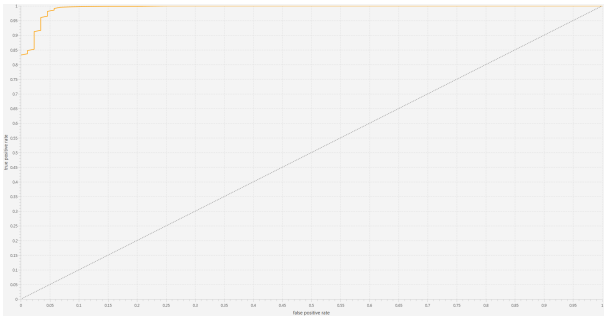


Non target class

Target class *8*



Target class

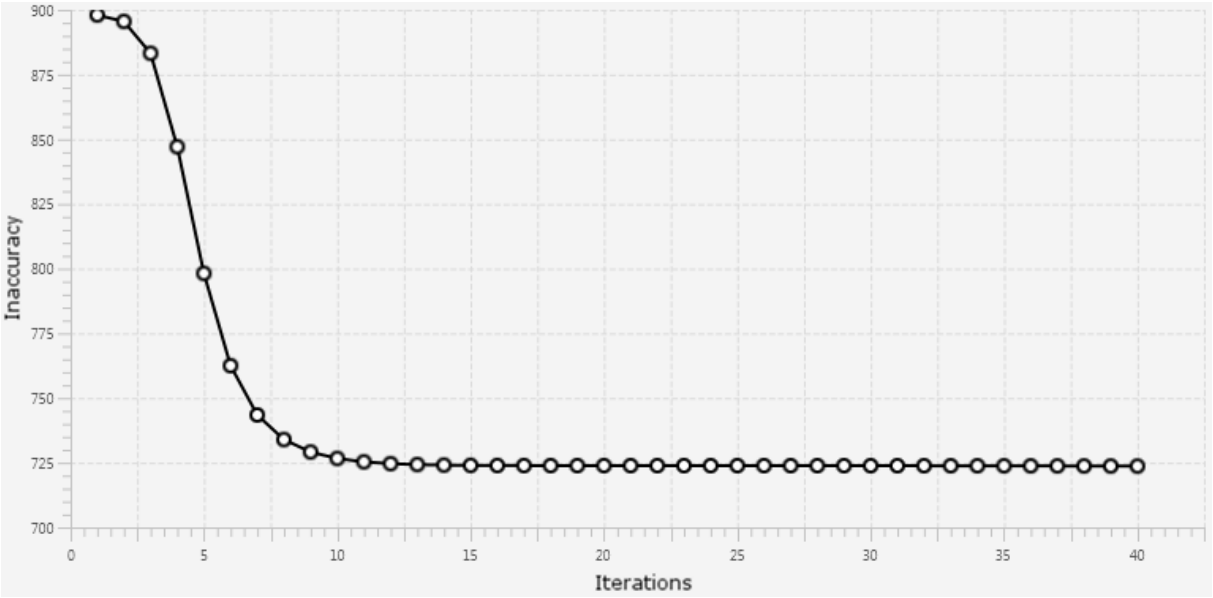


Non target class

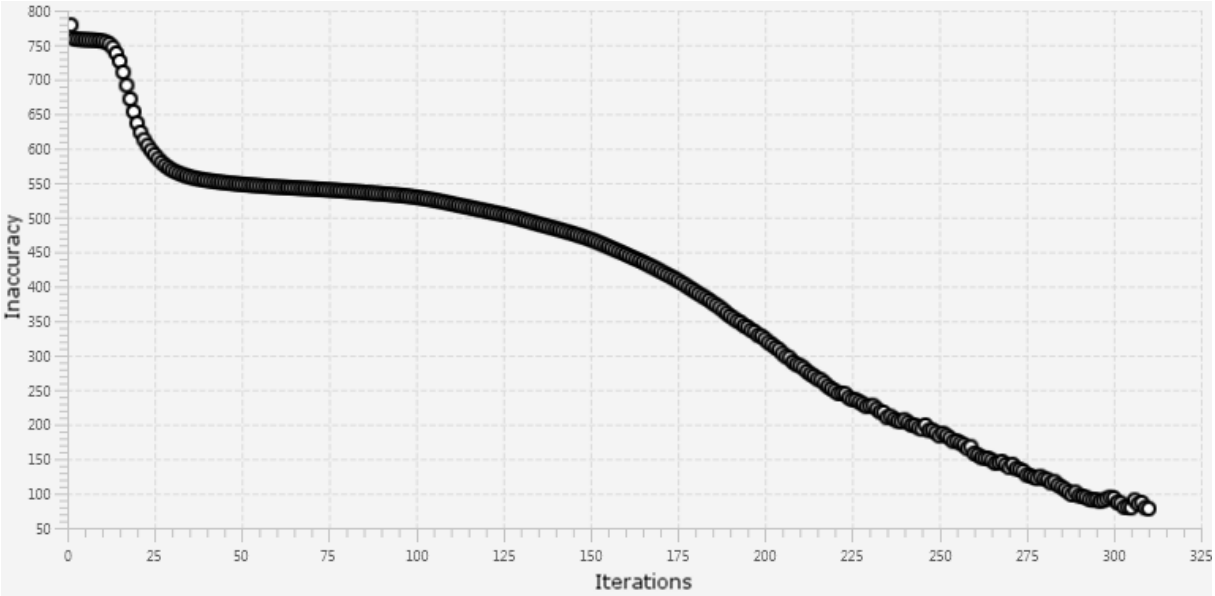
Target class *9*

ROC curves of the selected DANN model for the MNIST data set.

F Model Adjustments for the DANN Models Trained on the Artificial and German Credit Data Sets

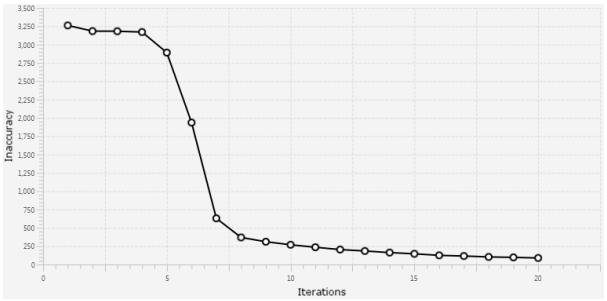


Model adjustment for the artificial data set.

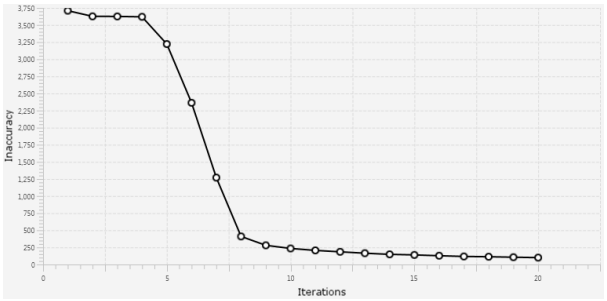


Model adjustment for the German credit data set.

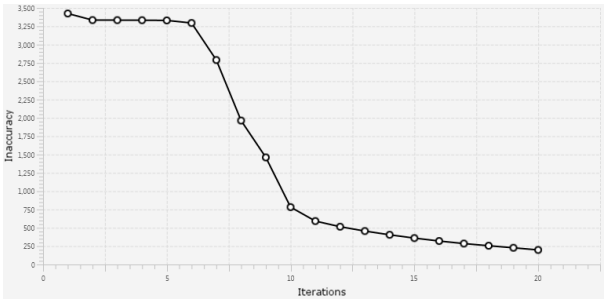
G Model Adjustment for the DANN Models Trained on the MNIST Database of Handwritten Digits



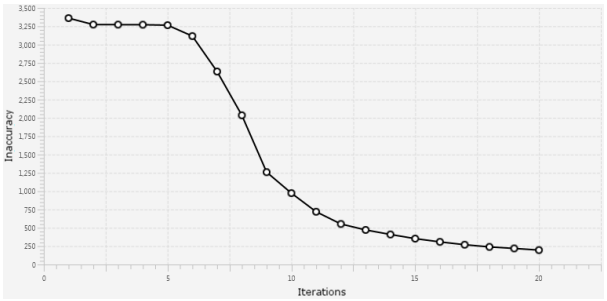
Target class *0*



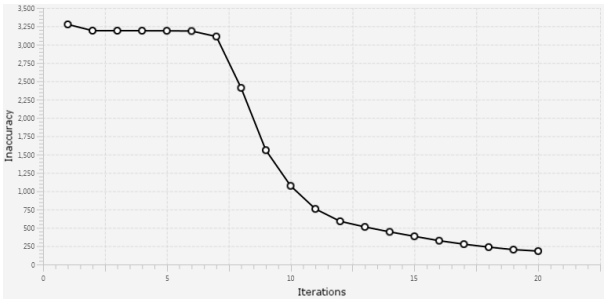
Target class *1*



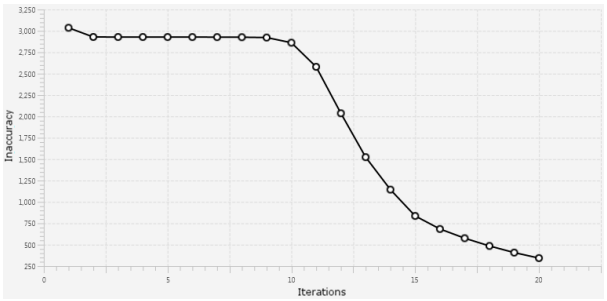
Target class *2*



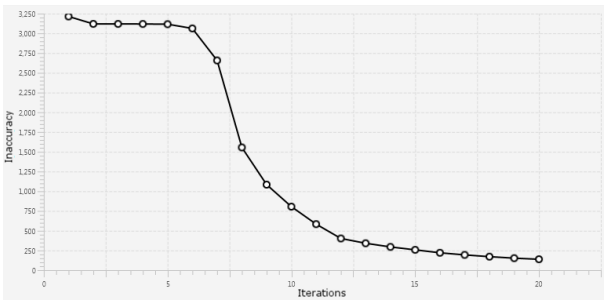
Target class *3*



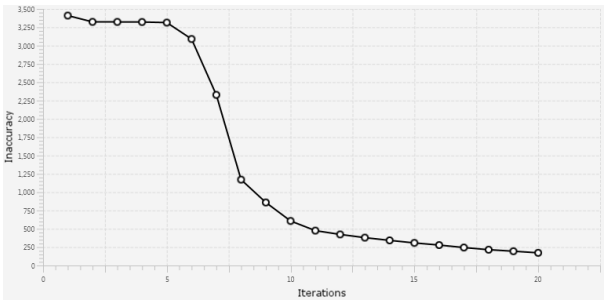
Target class *4*



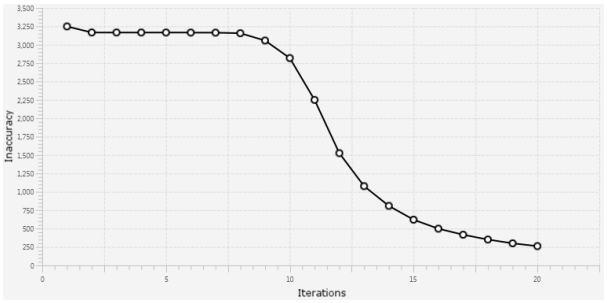
Target class *5*

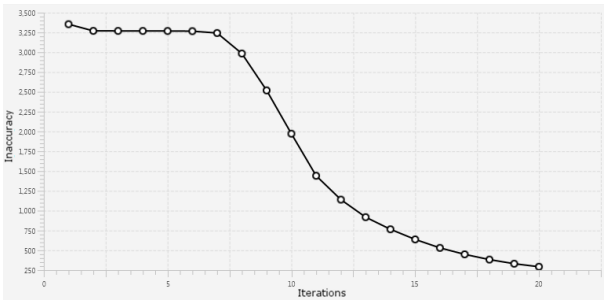


Target class *6*



Target class *7*





Target class *8* Target class *9*

Model adjustment for the MNIST data set.

H Measured results for the 10-fold cross validation training of the MNIST DANN models

Measure	Target class	Non target class	Weighted mean
AUC	0.9980936089984965	0.9980936089984964	0.9980936089984964
GINI	0.9961872179969928	0.9961872179969927	0.9961872179969928

Measured results for the 10-fold cross validation training of the DANN model used on the MNIST data set with the digit *0* as the target class.

Measure	Target class	Non target class	Weighted mean
AUC	0.998796342373518	0.9987963423735178	0.9987963423735179
GINI	0.9975926847470358	0.9975926847470356	0.9975926847470356

Measured results for the 10-fold cross validation training of the DANN model used on the MNIST data set with the digit *1* as the target class.

Measure	Target class	Non target class	Weighted mean
AUC	0.9921260185979422	0.9921260185979427	0.9921260185979426
GINI	0.9842520371958846	0.9842520371958852	0.9842520371958852

Measured results for the 10-fold cross validation training of the DANN model used on the MNIST data set with the digit *2* as the target class.

Measure	Target class	Non target class	Weighted mean
AUC	0.9943361476071346	0.9943361476071342	0.9943361476071342
GINI	0.9886722952142692	0.9886722952142684	0.9886722952142685

Measured results for the 10-fold cross validation training of the DANN model used on the MNIST data set with the digit *3* as the target class.

Measure	Target class	Non target class	Weighted mean
AUC	0.996345550869439	0.99634555086944	0.99634555086944
GINI	0.9926911017378878	0.9926911017378882	0.9926911017378882

Measured results for the 10-fold cross validation training of the DANN model used on the MNIST data set with the digit *4* as the target class.

Measure	Target class	Non target class	Weighted mean
AUC	0.9926475931992749	0.9926475931992752	0.9926475931992752
GINI	0.98525951863985498	0.98525951863985506	0.98525951863985504

Measured results for the 10-fold cross validation training of the DANN model used on the MNIST data set with the digit *5* as the target class.

Measure	Target class	Non target class	Weighted mean
AUC	0.995850667556361	0.995850667556356	0.995850667556356
GINI	0.9917013351127123	0.9917013351127116	0.9917013351127116

Measured results for the 10-fold cross validation training of the DANN model used on the MNIST data set with the digit *6* as the target class.

Measure	Target class	Non target class	Weighted mean
AUC	0.9938779297901568	0.9938779297901565	0.9938779297901565
GINI	0.9877558595803135	0.987755859580313	0.987755859580313

Measured results for the 10-fold cross validation training of the DANN model used on the MNIST data set with the digit *7* as the target class.

Measure	Target class	Non target class	Weighted mean
AUC	0.9910567895550149	0.9910567895550146	0.9910567895550146
GINI	0.9821135791100294	0.98211357911003	0.9821135791100295

Measured results for the 10-fold cross validation training of the DANN model used on the MNIST data set with the digit *8* as the target class.

Measure	Target class	Non target class	Weighted mean
AUC	0.9904080058921272	0.9904080058921267	0.9904080058921266
GINI	0.980816011784254	0.9808160117842535	0.9808160117842535

Measured results for the 10-fold cross validation training of the DANN model used on the MNIST data set with the digit *9* as the target class.

I Measured results for the 10-fold cross validation training of the MNIST DANN models

Measure	Target class	Non target class
Precision	0.9560439560439561	0.9988998899889989
Recall	0.9886363636363636	0.9956140350877193
F-Measure	0.9720670391061453	0.9972542559033497
Accuracy	0.995	0.995
AUC	0.9997383373205742	0.9997383373205742
GINI	0.9994766746411483	0.9994766746411483

Measured results for the selected DANN model and a decision threshold of 0.5 used on the MNIST data set with the digit *0* as the target class.

Measure	Target class	Non target class
Precision	0.9705882352941176	0.9977728285077951
Recall	0.9801980198019802	0.996662958843159
F-Measure	0.9753694581280787	0.9972175848636617
Accuracy	0.995	0.995
AUC	0.9995594665139484	0.9995594665139484
GINI	0.9991189330278969	0.9991189330278969

Measured results for the selected DANN model and a decision threshold of 0.5 used on the MNIST data set with the digit *1* as the target class.

Measure	Target class	Non target class
Precision	0.9484536082474226	0.9955703211517165
Recall	0.9583333333333334	0.9944690265486725
F-Measure	0.9533678756476685	0.9950193691200886
Accuracy	0.991	0.991
AUC	0.9992740597345133	0.9992740597345132
GINI	0.9985481194690267	0.9985481194690264

Measured results for the selected DANN model and a decision threshold of 0.5 used on the MNIST data set with the digit *2* as the target class.

Measure	Target class	Non target class
Precision	0.9553571428571429	0.9898648648648649
Recall	0.9224137931034483	0.994343891402715
F-Measure	0.9385964912280702	0.9920993227990971
Accuracy	0.986	0.986
AUC	0.9984006865345607	0.9984006865345609
GINI	0.9968013730691214	0.9968013730691219

Measured results for the selected DANN model and a decision threshold of 0.5 used on the MNIST data set with the digit *3* as the target class.

Measure	Target class	Non target class
Precision	0.927710843373494	0.9912758996728462
Recall	0.9058823529411765	0.9934426229508196
F-Measure	0.9166666666666667	0.99235807860262
Accuracy	0.986	0.986
AUC	0.9980070716811315	0.9980070716811316
GINI	0.9960141433622629	0.9960141433622631

Measured results for the selected DANN model and a decision threshold of 0.5 used on the MNIST data set with the digit *4* as the target class.

Measure	Target class	Non target class
Precision	0.8958333333333334	0.9966814159292036
Recall	0.9662921348314607	0.9890230515916575
F-Measure	0.9297297297297298	0.9928374655647383
Accuracy	0.987	0.987
AUC	0.9968425856263644	0.9968425856263644
GINI	0.993685171252788	0.993685171252788

Measured results for the selected DANN model and a decision threshold of 0.5 used on the MNIST data set with the digit *5* as the target class.

Measure	Target class	Non target class
Precision	0.99	0.9933333333333333
Recall	0.9428571428571428	0.9988826815642458
F-Measure	0.9658536585365853	0.9961002785515319
Accuracy	0.993	0.993
AUC	0.9996594839052939	0.9996594839052939
GINI	0.9993189678105878	0.9993189678105878

Measured results for the selected DANN model and a decision threshold of 0.5 used on the MNIST data set with the digit *6* as the target class.

Measure	Target class	Non target class
Precision	0.9292929292929293	0.9922308546059934
Recall	0.9292929292929293	0.9922308546059934
F-Measure	0.9292929292929293	0.9922308546059934
Accuracy	0.986	0.986
AUC	0.9981165708135742	0.9981165708135745
GINI	0.9962331416271484	0.996233141627149

Measured results for the selected DANN model and a decision threshold of 0.5 used on the MNIST data set with the digit *7* as the target class.

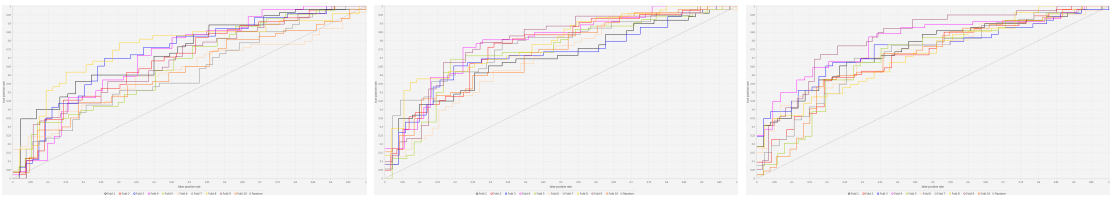
Measure	Target class	Non target class
Precision	0.9680851063829787	0.9911699779249448
Recall	0.9191919191919192	0.9966703662597114
F-Measure	0.9430051813471503	0.9939125622578859
Accuracy	0.989	0.989
AUC	0.9970066929001447	0.9970066929001445
GINI	0.9940133858002893	0.9940133858002891

Measured results for the selected DANN model and a decision threshold of 0.5 used on the MNIST data set with the digit *8* as the target class.

Measure	Target class	Non target class
Precision	0.9866666666666667	0.985945945945946
Recall	0.8505747126436781	0.9989047097480832
F-Measure	0.9135802469135803	0.9923830250272035
Accuracy	0.986	0.986
AUC	0.9943598846797849	0.994359884679785
GINI	0.9887197693595697	0.98871976935957

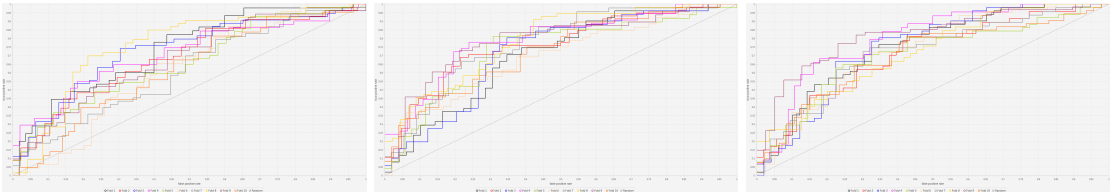
Measured results for the selected DANN model and a decision threshold of 0.5 used on the MNIST data set with the digit *9* as the target class.

J ROC Curves for the Logistic Regression Models on the Influence Data Sets



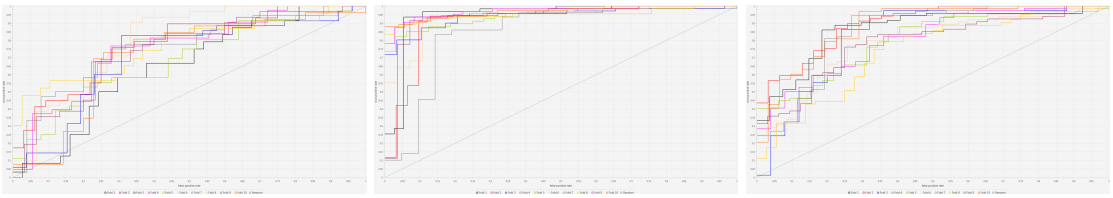
LICON GSA PLAY

ROC curves of the logistic regression model for the target class trained on the influence data set build by each method for the artificial data set.



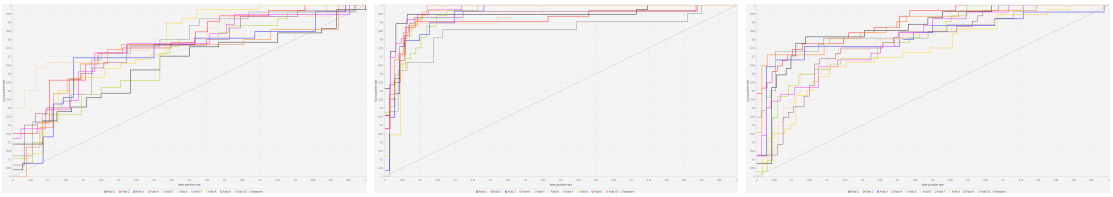
LICON GSA PLAY

ROC curves of the logistic regression model for the non target class trained on the influence data set build by each method for the artificial data set.



LICON GSA PLAY

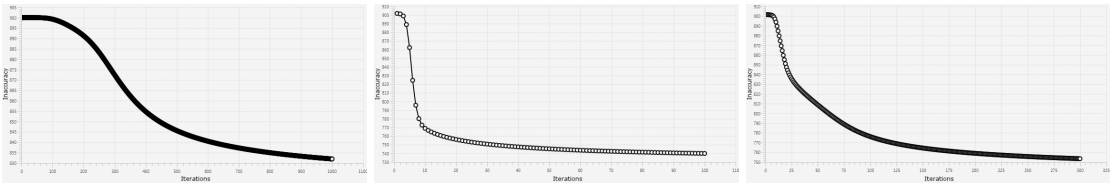
ROC curves of the logistic regression model for the target class trained on the influence data set build by each method for the German credit data set.



LICON GSA PLAY

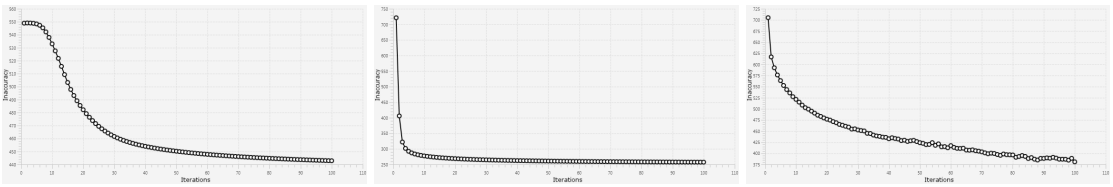
ROC curves of the logistic regression model for the non target class trained on the influence data set build by each method for the German credit data set.

K Model Adjustments for the Logistic Regression Models on the Influence Data Sets



LICON GSA PLAY

Model adjustment for the logistic regression model trained on the influence data set build by each method for the artificial data set.

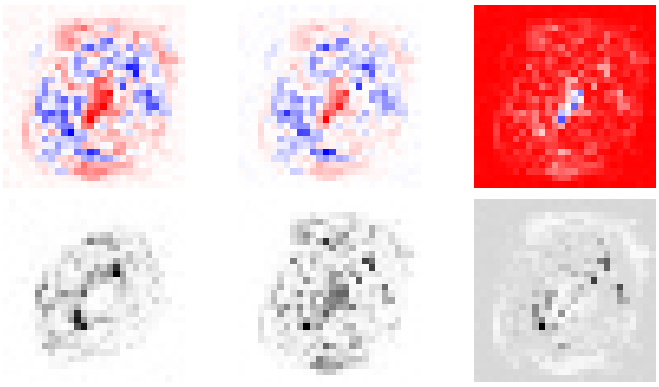


LICON GSA PLAY

Model adjustment for the logistic regression model trained on the influence data set build by each method for the German credit data set.

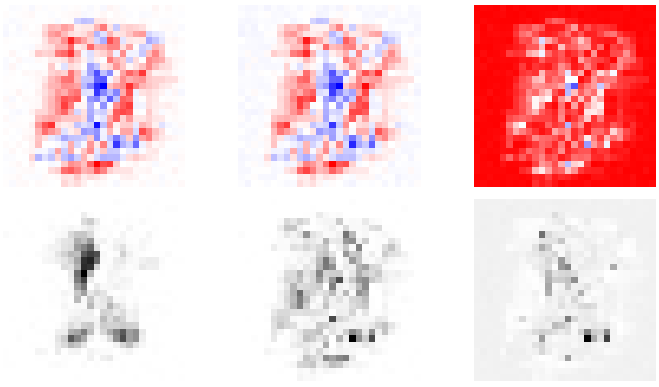
L Mean and Variance Images for the MNIST Data Sets

For the images showing the mean influence values, blue colored pixels indicate a positive influence and red colored pixels display a negative influence. For the images showing the variance of in the influence values, black indicates the highest and white the lowest variance.



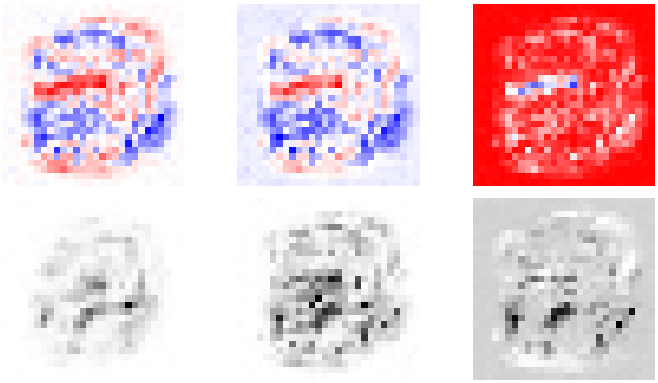
LICON PLAY GSA

Mean (top) and variance (bottom) of all influences calculated with the MNIST data set for the digit 0.



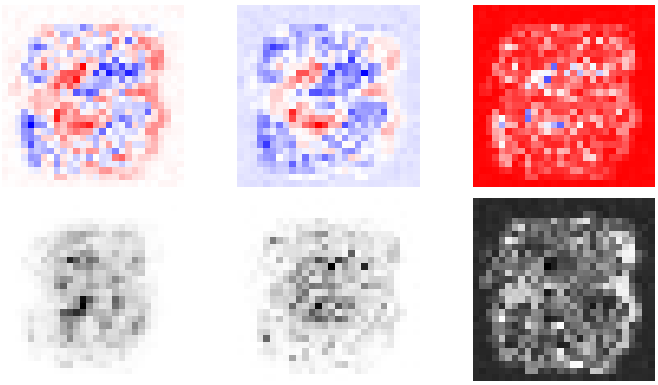
LICON PLAY GSA

Mean (top) and variance (bottom) of all influences calculated with the MNIST data set for the digit 1.



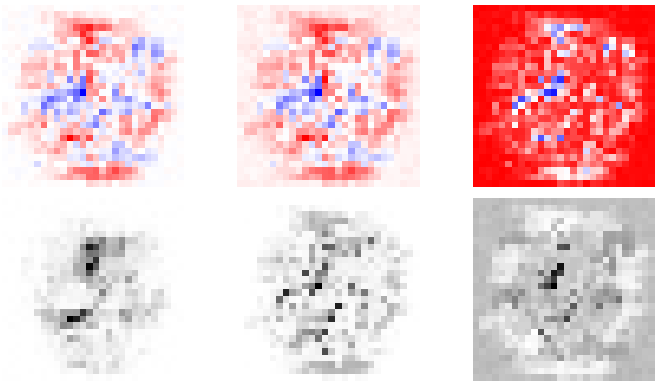
LICON PLAY GSA

Mean (top) and variance (bottom) of all influences calculated with the MNIST data set for the digit 2.



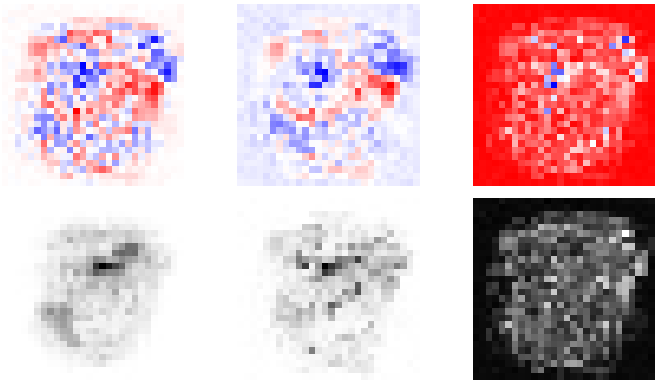
LICON PLAY GSA

Mean (top) and variance (bottom) of all influences calculated with the MNIST data set for the digit 3.



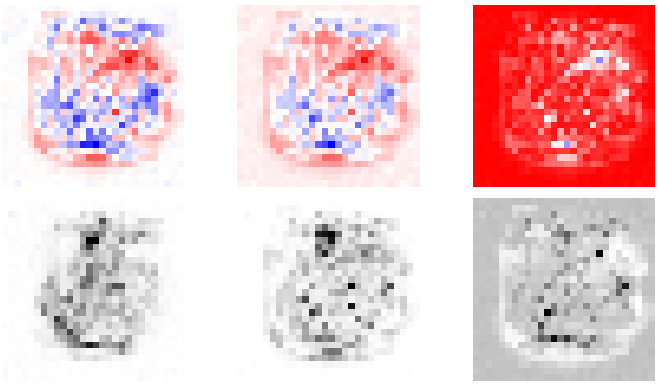
LICON PLAY GSA

Mean (top) and variance (bottom) of all influences calculated with the MNIST data set for the digit 4.



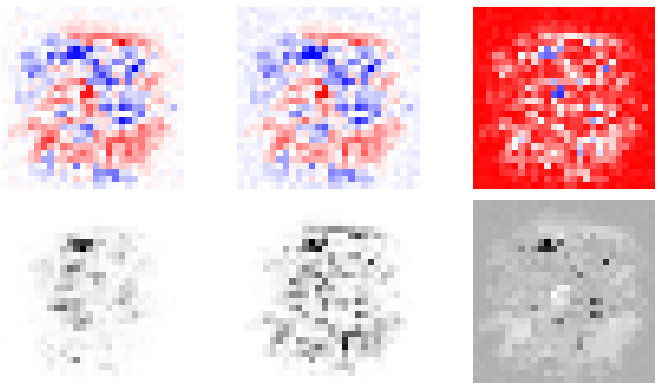
LICON PLAY GSA

Mean (top) and variance (bottom) of all influences calculated with the MNIST data set for the digit 5.



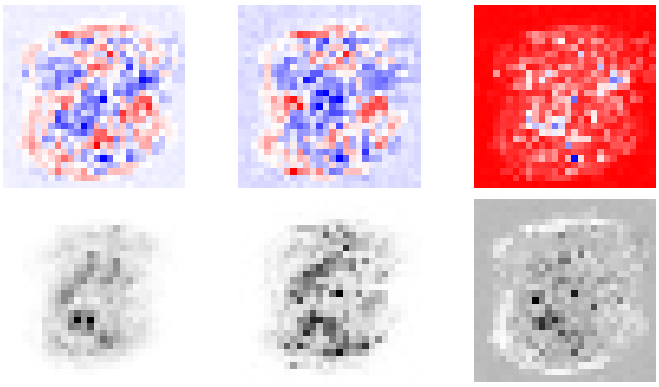
LICON PLAY GSA

Mean (top) and variance (bottom) of all influences calculated with the MNIST data set for the digit 6.



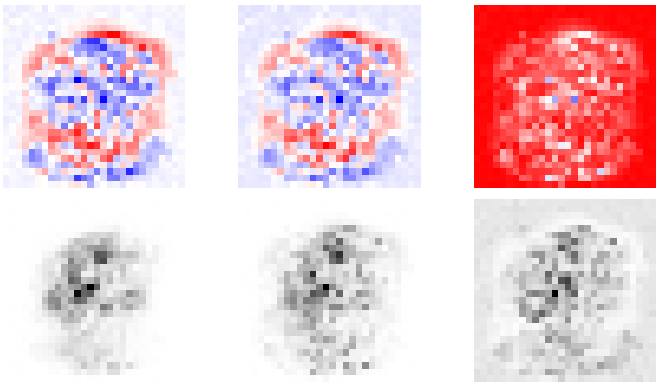
LICON PLAY GSA

Mean (top) and variance (bottom) of all influences calculated with the MNIST data set for the digit 7.



LICON PLAY GSA

Mean (top) and variance (bottom) of all influences calculated with the MNIST data set for the digit 8.



LICON PLAY GSA

Mean (top) and variance (bottom) of all influences calculated with the MNIST data set for the digit 9.

References

Bach, S. et al. (2015). 'On Pixel-wise Explanations for Non-Linear Classifier Decisions by Layer-wise Relevance Propagation'. In: *PLoS ONE* 10.7, pp. 1–44.

Baehrens, D. et al. (2010). 'How to Explain Individual Classification Decisions'. In: *Journal of Machine Learning Research* 11.Jun, pp. 1803–1831.

Benitez, J. M., Castro, J. L., and Requena, I. (1997). 'Are Artificial Neural networks Black Boxes?' In: *IEEE Transactions on Neural Networks* 8.5, pp. 1156–1164.

Biggam, J. (2015). *Succeeding with your Master's Dissertation: A step-by-step handbook*. 3rd edn. Berkshire, England: Open University Press.

Cao, M. and Qiao, P. (2008). 'Neural network committee-based sensitivity analysis strategy for geotechnical engineering problems'. In: *Neural Computing and Applications* 17.5–6, pp. 509–519.

Cortez, P. and Embrechts, M. J. (2011). 'Opening Black Box Data Mining Models Using Sensitivity Analysis'. In: *2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, pp. 341–348. DOI: 10.1109/CIDM.2011.5949423.

– (2013). 'Using sensitivity analysis and visualization techniques to open black box data mining models'. In: *Information Sciences* 225, pp. 1–17.

Crawford, K. and Calo, R. (2016). 'There is a blind spot in AI research'. In: *Nature* 538.7625, pp. 311–313.

Defence Advanced Research Projects Agency (2016). *Broad Agency Announcement - Explainable Artificial Intelligence (XAI)*. http://www.darpa.mil/attachments/DARPA-BAA-16-53.pdf. Online. Accessed 01 December 2017.

Dua, D. and Karra Taniskidou, E. (2017). *UCI Machine Learning Repository*. Online. Accessed 16 September 2018. URL: https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data).

Embrechts, M., Ozdemir, F. A. M., and Kewley, R. (2003). 'Data mining for molecules with 2-D neural network sensitivity analysis'. In: *International Journal of smart engineering system design* 5.4, pp. 225–239.

Engelbrecht, A. P., Cloete, I., and Zurada, J. M. (1995). 'Determining the significance of input parameters using sensitivity analysis'. In: *From Natural to Artificial Neural Com-*

putation: International Workshop on Artificial Neural Networks Malaga-Torremolinos, Spain, June 7–9, 1995 Proceedings. Ed. by Mira, J. and Sandoval, F. Berlin, Germany: Springer, pp. 382–388. ISBN: 978-3-540-49288-7. URL: https://doi.org/10.1007/3-540-59497-3_199.

Firth-Butterfield, K. (5th Feb. 2017). 'AI can win at poker: but as computers get smarter, who keeps tabs on their ethics?' In: *The Guardian*. Online. Accessed 28 November 2017. URL: https://www.theguardian.com/technology/2017/feb/05/artificial-intelligence-ethics-poker-libratus-texas-holdem-ai-deepstack.

Fischer, A. (2015). 'How to determine the unique contributions of input-variables to the nonlinear regression function of a multilayer perceptron'. In: *Ecological Modelling* 309-310, pp. 60–63.

Frank, E., Hall, M. A., and Witten, I. H. (2016). *The WEKA Workbench*. Ed. by Kaufmann, M. 4th edn. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques". URL: https://www.cs.waikato.ac.nz/ml/weka/index.html.

Garson, G. D. (1991). 'Interpreting neural network connection weights.' In: *Artificial Intelligence Expert* 6, pp. 47–51.

Gevrey, M., Dimopoulos, I., and Lek, S. (2003). 'Review and comparison of methods to study the contribution of variables in artificial neural network models'. In: *Ecological Modelling* 160, pp. 249–264.

Giam, X. and Olden, J. D. (2015). 'A new R^2-based metric to shed greater insight on variable importance in artificial neural networks'. In: *Ecological Modelling* 313, pp. 307–313.

Goh, A. T. C. (1995). 'Back-propagation neural networks for modeling complex systems.' In: *Artificial Intelligence in Engineering* 9, pp. 143–151.

Green, M. et al. (2009). 'Exploring new possibilities for case-based explanation of artificial neural network ensembles'. In: *Neural Networks* 22.1, pp. 75–81.

Hinton, G. et al. (2012). 'Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups'. In: *Signal Processing Magazine, IEEE* 29.6, pp. 82–97.

Ibrahim, O. M. (2013). 'A comparison of methods for assessing the relative importance of input variables in artificial neural networks'. In: *Journal of Applied Sciences Research* 9.11, pp. 5692–5700.

John, G. H., Kohavi, R., and Pfleger, K. (1994). 'Irrelevant Features and the Subset Selection Problem'. In: *Machine Learning: Proceedings of the Eleventh International Conference*. Ed. by Cohen, W. W. and Hirsh, H. San Francisco, CA: Morgan Kaufmann, pp. 121–129.

Kasneci, G. and Gottron, T. (2016). '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 2016 10. Indianapolis, Indiana, USA: ACM, pp. 45–54. ISBN: 978-1-4503-4073-1. DOI: 10.1145/2983323.2983746. URL: http://doi.acm.org/10.1145/2983323.2983746.

Kemp, S. J., Zaradic, P., and Hansen, F. (2007). 'An approach for determining relative input parameter importance and significance in artificial neural networks'. In: *Ecological Modelling* 204.3–4, pp. 326–334.

Kewley, R. H., Embrechts, M. J., and Breneman, C. (2000). 'Data strip mining for the virtual design of pharmaceuticals with neural networks'. In: *IEEE Transactions on Neural Networks* 11.3, pp. 668–679.

Keysers, D. (2007). 'Comparison and Combination of State-of-the-art Techniques for Handwritten Character Recognition: Topping the MNIST Benchmark'. In: *ArXiv e-prints*. arXiv: 0710.2231.

Kindermans, P.-J. et al. (2017a). 'Learning how to explain neural networks: PatternNet and PatternAttribution'. In: *ArXiv e-prints*. arXiv: 1705.05598 [stat.ML].

Kindermans, P.-J. et al. (2017b). 'The (Un)reliability of saliency methods'. In: *ArXiv e-prints*. arXiv: 1711.00867 [stat.ML].

Klimasauskas, C. C. (1991). 'Neural Nets Tell Why: A technique for explaining a neural networkś decision-making process'. In: *Dr. Dobbś Journal* 16.4, pp. 16–25.

Imagenet classification with deep convolutional neural networks (2012), pp. 1097–1105.

Lek, S. et al. (1996). 'Role of some environmental variables in trout abundance models using neural networks'. In: *Aquatic Livin Resources* 9.1, pp. 23–29.

Lin, Y. and Cunningham, G. A. (1995). 'A new Approach to Fuzzy-Neural System Modeling'. In: *IEEE Transactions on Fuzzy Systems* 3.2, pp. 190–198.

Lipton, Z. C. (2016). 'The Mythos of Model Interpretability'. In: *ArXiv e-prints*. arXiv: 1606.03490 [cs.LG].

Mak, B. and Blanning, R. W. (1998). 'An Empirical Measure of Element Contribution in Neural Networks'. In: *IEEE Transactions on systems, man, and cybernetics - Part C: Applications and Reviews* 28.4, pp. 561–564.

Feature Selection Using Neural Networks with Contribution Measures (1995). Canberra, Australia, pp. 124–136.

Minh, V. et al. (2015). 'Human-level control trough deep reinforcement learning.' In: *Nature* 518.7540, pp. 529–533.

Mizukami, Y. et al. (2010). 'CUDA Implementation of Deformable Pattern Recognition and its Application to MNIST Handwritten Digit Database'. In: *2010 20th International Conference on Pattern Recognition*, pp. 2001–2004. DOI: 10.1109/ICPR.2010.493.

Montano, J. J. and Palmer, A. (2003). 'Numeric sensitivity analysis applied to feedforward neural networks'. In: *Neural Computing and Applications* 12.2, pp. 119–125. ISSN: 0941-0643. URL: https://doi.org/10.1007/s00521-003-0377-9.

Montavon, G., Samek, W., and Müller, K.-R. (2018). 'Methods for interpreting and understanding deep neural networks'. In: *Digital Signal Processing* 73, pp. 1–15. URL: https://doi.org/10.1016/j.dsp.2017.10.011.

Montavon, G. et al. (2017). 'Explaining nonlinear classification decisions with deep Taylor decomposition'. In: *Pattern Recognition* 65, pp. 211–222. URL: https://doi.org/10.1016/j.patcog.2016.11.008.

Olden, J. D. and Jackson, D. A. (2002). 'Illuminating the "black box": a randomization approach for understanding variable contributions in artificial neural networks'. In: *Ecological Modelling* 154.1–2, pp. 135–150.

Olden, J. D., Joy, M. K., and Death, R. G. (2004). 'An accurate comparison of methods for quantifying variable importance in artificial neural networks using simulated data'. In: *Ecological Modelling* 178.3–4, pp. 389–397.

Ona, J. de and Garrido, C. (2014). 'Extracting the contribution of independent variables in neural network models: a new approach to handle instability'. In: *Neural Computing and Applications* 25.3–4, pp. 859–869.

OpenAI (2017). *Dota 2*. Online. Accessed 06 December 2017. URL: https://blog.openai.com/dota-2/.

Paliwal, M. and Kumar, U. A. (2011). 'Assessing the contribution of variables in feed forward neural network'. In: *Applied Soft Computing* 11.4, pp. 3690–3696.

Papadokonstantakis, S., Lygeros, A., and Jacobsson, S. P. (2006). 'Comparison of recent methods for inference of variable influence in neural networks'. In: *Neural Networks* 19.4, pp. 500–513.

Pentos, K. (2016). 'The methods of extracting the contribution of variables in artificial neural network models - Comparison of inherent instability'. In: *Computers and Electronics in Agriculture* 127, pp. 141–146.

Ploeg, S. van der (2010). *Bank Default Prediction Models: A Comparison and an Application to Credit Rating Transitions*. Rotterdam.

Reddy, N. S. et al. (2015). 'Artificial neural network modeling on the relative importance of alloying elements and heat treatment temperature to the stability of α and β phase in titanium alloys'. In: *Computational Materials Science* 107, pp. 175–183.

Reed, R. (1993). 'Pruning Algorithms - A survery'. In: *IEEE Transactions on Neural Networks* 4.5, pp. 740–747.

Rohrer, M. W. (2000). 'Seeing is believing: the importance of visualization in manufacturing simulation'. In: *2000 Winter Simulation Conference Proceedings (Cat. No.00CH37165)*. Vol. 2, pp. 1211–1216.

Saltelli, A. et al. (2008). *Global Sensitivity Analysis: The Primer*. West Sussex, England: John Wiley and Sons Ltd.

Samek, W., Wiegand, T., and Müller, K.-R. (2017). 'Explainable Artificial Intelligence: Understanding, Visualizing and Interpreting Deep Learning Models'. In: *ITU Journal: ICT Discoveries* 1, pp. 1–10.

Sample, I. (2017). 'It's able to create knowledge itself': Google unveils AI that learns on its own'. In: *The Guardian*. Online. Accessed 01 December 2017. URL: https://www.theguardian.com/science/2017/oct/18/its-able-to-

create-knowledge-itself-google-unveils-ai-learns-all-on-its-own.

Schulte-Mattler, H., Daun, U., and Manns, T. (2004). 'Basel II: Trennschärfemaß zur Validierung von internen Rating-Systemen'. In: *RATINGaktuell* 2004.6, pp. 46–52.

Setiono, R., Baesens, B., and Mues, C. (2008). 'Recursive Neural Network Rule Extraction for Data With Mixed Attributes'. In: *IEEE Transactions on Neural Networks* 19.2, pp. 299–307.

Simard, P. Y., Steinkraus, D., and Platt, J. (2003). 'Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis'. In: Institute of Electrical and Electronics Engineers, Inc. URL: https://www.microsoft.com/en-us/research/publication/best-practices-for-convolutional-neural-networks-applied-to-visual-document-analysis/.

Simonyan, K., Vedaldi, A., and Zisserman, A. (2014). 'Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps'. In: *ArXiv e-prints*. arXiv: 1312.6034v2 [cs.CV].

Smilkov, D. et al. (2017). 'SmoothGrad: removing noise by adding noise'. In: *ArXiv e-prints*. arXiv: 1706.03825v1 [cs.LG].

Sokolova, M., Japkowicz, N., and Szpakowicz, S. (2006). 'Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation'. In: *AI 2006: Advances in Artificial Intelligence*. Ed. by Sattar, A. and Kang, B. ho. Heidelberg: Springer, pp. 1015–1021.

Springenberg, J. T. et al. (2015). 'Striving for Simplicity: The All Convolutional Net'. In: *ArXiv e-prints*. arXiv: 1412.6806v3 [cs.LG].

Sundararajan, M., Taly, A., and Yan, Q. (2017). 'Axiomatic Attribution for Deep Networks'. In: *ArXiv e-prints*. arXiv: 1703.01365v2 [cs.LG].

Sung, A. H. (1998). 'Ranking importance of input parameters of neural networks'. In: *Expoert Systems with Applications* 15.3–4, pp. 405–411.

Tchaban, T., Taylor, M. J., and Griffin, J. P. (1998). 'Establishing impacts of the inputs in a feedforward neural network'. In: *Neural Computing and Applications* 7.4, pp. 309–317.

Tzeng, F.-Y. and Ma, K.-L. (2005). 'Opening the Black Box - Data Driven Visualization of Neural Networks'. In: Minneapolis, MN, USA, pp. 383–390.

Y. LeCun, C. C. and Burges, C. J. (1998). *The mnis database of handwritten digits*. Online. Accessed 16 September 2018. URL: http://yann.lecun.com/exdb/mnist/.

Yeh, I.-C. and Cheng, W.-L. (2010). 'First and second order sensitivity analysis of MLP'. In: *Neurocomputing* 73.10–12, pp. 2225–2233.

Zeiler, M. D. and Fergus, R. (2014). 'Visualizing and Understanding Convolutional Networks'. In: *Computer Vision – ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*. Ed. by Fleet, D. et al. Cham: Springer International Publishing, pp. 818–833.

Sensitivity analysis for minimization of input data dimension for feedforward neural network (1994). Vol. 6, pp. 447–450.

Erklärung zur Abschlussarbeit gemäß § 23 Abs. 7 APB der TU Darmstadt

Hiermit versichere ich, Torsten Dietl, die vorliegende Master-Thesis ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Mir ist bekannt, dass im Fall eines Plagiats (§ 38 Abs. 2 APB) ein Täuschungsversuch vorliegt, der dazu führt, dass die Arbeit mit 5,0 bewertet und damit ein Prüfungsversuch verbraucht wird. Abschlussarbeiten dürfen nur einmal wiederholt werden.

Bei der abgegebenen Thesis stimmen die schriftliche und die zur Archivierung eingereichte elektronische Fassung überein.

Bei einer Thesis des Fachbereichs Architektur entspricht die eingereichte elektronische Fassung dem vorgestellten Modell und den vorgelegten Plänen.

Datum:	Unterschrift:
02. Oktober 2018	Torsten Dietl