Interpretability of deep neural networks

Juri Zach
Hamburg University of Applied Sciences, Department of Computer Science
Hamburg, Germany
juri.zach@haw-hamburg.de

***Abstract*—This paper gives an entry point to the problem of interpreting machine learning models. It includes an introduction to machine learning models and their complexity, followed by general aspects of interpretability. The focus, however, is put on deep neural networks and modern techniques to increase their interpretability.**

***Index Terms*—machine learning, deep neural networks, convolutional network, activation maximization, feature inversion**

I. Introduction

The Research of Artificial Intelligence goes back to 1950 when Alan Turing introduced the Turing Test to determine the existence of Artificial Intelligence [1]. At this point in time the expectations for Artificial Intelligence were extremely high. After it became clear that the expectations cannot be matched, the subject was avoided for many years. Recently the field of Artificial Intelligence gained enormous interest again. The main reason for this are great successes in machine learning due to growing computational powers and big data. Machine learning algorithms could be trained on huge datasets, sometimes exceeding the performance of humans in special fields of interest such as playing chess or Go. Especially complex machine learning models such as deep neural networks impress with very high predictive accuracy. However, because of their complexity, these machine learning algorithms are hard to interpret and their decisions cannot be explained. This causes a number of problems which need to be dealt with, if Artificial Intelligence is to become part of our modern society by supporting humans with difficult and important decisions.

This paper gives an overview about the interpretability of machine learning in general and focuses on deep neural networks in specific. Section II gives an overview on machine learning algorithms and a description of deep neural networks. Section III covers the current state of science and technology in interpreting machine learning, including its definition, importance and evaluation of interpretability, while section IV describes general methods of interpretation.
Sections V, VI and VII deal with modern and promising algorithms to interpret deep neural networks.

II. Machine learning

Machine learning is a branch of Artificial Intelligence that deals with the appliance of algorithms on huge amounts of data to improve the performance of a given task through experience and training. In the training process a machine learning algorithm characterizes underlying relationships within large arrays or matrices to recognize patterns.
Common fields of application are: Extracting of information from huge amounts of data (data mining), classification, regression tasks or predicting of the future from information from the past.
[2]

Machine learning can be separated into three main types of learning: Supervised learning, unsupervised learning and reinforcement learning.

- **Supervised learning** is using labeled training data to synthesize a model function, generalizing the relationship between feature Vectors and the supervisor labels. If the model is trained well, it can accurately predict class labels for unobserved data instances.
- **Unsupervised learning** algorithms discover structures in unlabeled datasets. For this kind of learning it is not necessary to know the desired target output.
- **Reinforcement learning** is a learning technique for agents to explore an environment. The agent is executing an adaptive sequence of actions and observes the response of the environment state. To improve the behavior within the environment, a reward function is being maximized. [2]

Machine learning is a generic term, including a wide range of different algorithms or models to solve learning problems. Every model has its own advantages and disadvantages, depending on the way of learning and many other properties. Very popular machine learning models are: Decision trees, linear models, logistic regression, Bayesian nets, random forest and deep neural networks.

To build a machine learning application, an algorithm is trained with data, containing the problem information. The machine learning algorithm extracts the information from the data and builds a model that represents the learned knowledge. This model can be used to solve a given problem such as classification or prediction. How the model remembers the learned knowledge, depends on the kind of machine learning method that was used.

A. Complexity of machine learning algorithms

The ability to learn from data, given a suitable dataset, depends on the machine learning algorithm that is used. Experience shows, that the most successful machine learning

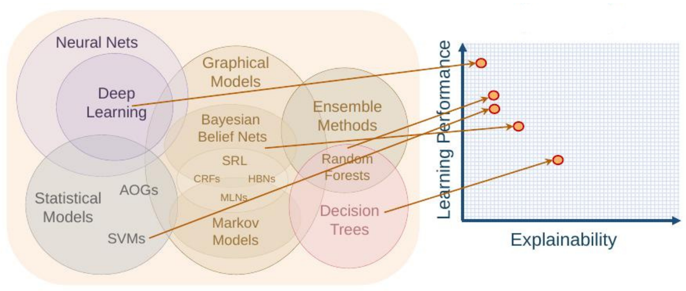


Fig. 1. Learning performance vs explainability of different machine learning models. [4]

algorithms are also the most complex. Unfortunately, with the complexity of a model comes the loss of insight and therefore an inability to explain the model's behavior. Understanding the information a model extracts from the training data and how the model solves a given task is hidden within the parameters of complex models.
A decision tree for example is a fairly simple model. It is very useful for representing knowledge, because it can easily be read by humans. However, its learning abilities are limited. Deep neural networks in comparison are the most successful machine learning algorithms in modern research. This technique has reached extremely high predictive accuracy, sometimes exceeding human performances. However, a deep neural network is described by up to millions of different values and does not give insight into its knowledge or the way decisions are made.

Because of their relevance for artificial intelligence, this paper will focus on interpretation methods for deep neural networks, which are introduced in section II-B. Because nearly the entire research on interpretation methods for deep neural networks is concerning the field of image recognition, deep convolutional networks are explained as well. They are the preferred deep learning solutions for image classifications and are used as an example network in later sections.

B. Deep neural networks

Inspired by the human brain, artificial neural networks consist of many simple processors which are called neurons. The functionality of a single neuron is shown in figure 2. Each neuron is connected to input sensors or other neurons, processing their weighted signals to produce an activation signal. Neural networks often combine thousand of neurons with millions of connections between these neurons. Typically they are structured in different layers, where each neuron of a layer is connected to all or a partition of the previous layers. A traditionally used layer is the fully connected layer, in which every neuron of this layer is connected to all neurons of the previous layers. Specialized layers such as convolutional and pooling layers are explained in section II-C. The difference between a shallow and a deep neural network consists in the number of layers.

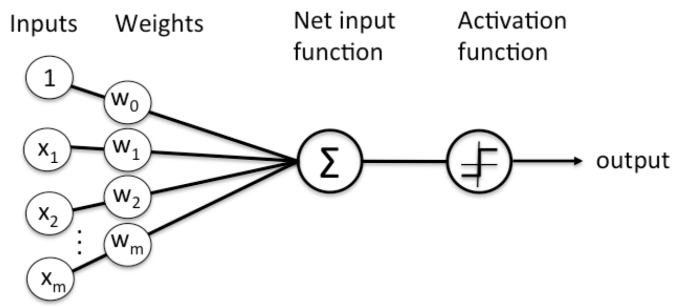
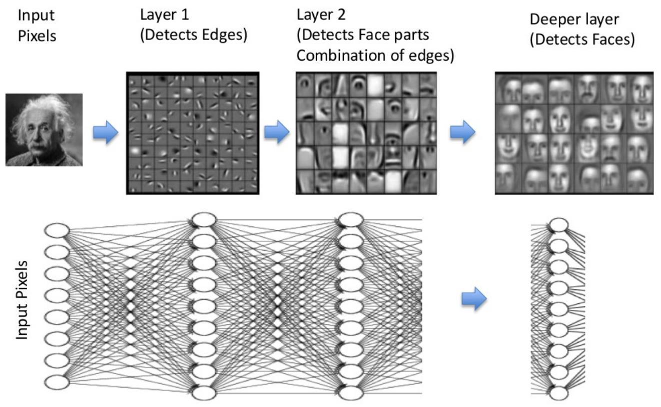


Fig. 2. Inspired by neurons of the human brain, an artificial neuron sums weighted inputs and used an activation function to produce a output. [19]



Input Pixels
Layer 1 (Detects Edges)
Layer 2 (Detects Face parts Combination of edges)
Deeper layer (Detects Faces)
Input Pixels

Fig. 3. Example for representation learning of deep neural networks. In the first layers the neural network learns features like edges which are combined to abstract objects in later layers. [20]

Due to their complexity, deep neural networks succeed in central artificial intelligence problems where other machine learning algorithms fail [7, p.151]. The difficulty of problems, such as speech recognition and object detection, results in their high dimensional input. With an exceeding number of variables the possible distinct configuration of these variables increase exponentially, often transcending the number of training examples. This kind of problem, also referred to as the curse of dimensionality, exceeds the learning capability of many traditional machine learning methods such as k-means, k-nearest neighbors, decision trees or n-gramm models. The reason for this is, that traditional machine learning methods learn symbolic representation. Deep learning methods such as deep neural networks meet the curse of dimensionality by storing knowledge in distributed representations.
Compared to symbolic representations, distributed representations do not store learned knowledge in one feature, but in many distributed features which describe partial knowledge on different levels. In deep neural networks, the first layers learn low level features, while intermediate layers combine them to higher level features.
This process is visualized on an image classifier, which is trained to recognize faces, in figure 3. [7]

Depending on the layers configuration neural networks are differentiated in feed forward neural networks and recurrent neural networks. Under the assumption that earlier inputs do not have an effect on the current output, signals of feed forward neural networks are only passed forward in one direction. Recurrent neural networks however, use their output of previous steps to compute the current step, which makes them the deepest of all neural networks.
Section II-C explains convolutional neural networks which are a specialized kind of feed forward neural networks. They are widely used in the domain of computer vision and will serve as an example network for interpretation methods in later sections.

C. *Convolutional neural network*

A convolutional neural network is a network architecture specialized for grid like typology, such as time series, that can be thought of one dimensional grids, or images as two dimensional grids. The core elements of a convolutional neural network are convolutional layers, often combined with pooling layers and fully connected layers.

A convolutional layer implements several filters, each specialized to detect a specific pattern. Technically a filter is a small matrix which is slit over the feature matrix of the previous layer to calculate the weighted inputs of a specific neuron. Because the filter matrix is smaller than the feature matrix, a filter is able to combine spatially close located input features to one, more meaningful, output feature.
Sharing the filter parameters over all input features enables a filter to detect meaningful relations in multiple locations and reduces the memory and computational requirements. By combining several filters, a convolutional layer detects a number of meaningful features and their location.
A pooling layer is often applied after the convolutional layer. Its task is to provide summary statistics of nearby outputs by averaging them or taking the maximum value. More detailed and mathematical information on convolutional neural networks are provided by Ian Goodfellow [7].

III. Theory of Interpretability

This section provides the theoretical basics of machine learning interpretability. III-A explains the meaning of interpretability while section III-B points out the importance of this subject. Furthermore, section III-C gives some ideas on how the interpretability can be evaluated.

A. *What is interpretability?*

Although there are many papers that claim to implement a interpretable model, there does not seem to be an established definition for the inpterpretability of machine learning. ”The term interpretability is ill-defined, and thus claims regarding interpretability of various models may exhibit a quasi-scientific character” [6].

Definitions of interpretability vary, depending on the different papers and experts and seems to remain elusive. In this paper a fairly simple definition is used: ”we define interpretability as the ability to explain or present in understandable therms to humans” [3]. The ability to explain is understood as the ability to reason a single prediction of a machine learning decision on one input example. The ability to present in understandable form to humans is understood as the ability to communicate the information that was extracted from data by a machine learning algorithm to humans in an understandable way.

B. *Importance of interpretable machine learning*

The importance of interpretable machine learning algorithm comes from an incompleteness of the problem formalization [3]. Tasks that can be solved by machine learning algorithms cannot be formulated in a reasonable number of rules, but must be learned actively by a machine. The programmer often lacks understanding of the task itself and has limited insight into the knowledge learned by the machine. Interpretability can help the programmer and users to gain additional knowledge about the problem and to develop trust in the machine learning model. In the following, the most important reasons why interpretable machine learning algorithm are needed are listed.

- **Debugging** is especially difficult for most machine learning algorithms, due to their black box behavior. Most applications are evaluated by their accuracy. However it is often unknown how accuracy can be improved. Problems can be a bad quality of the training data, the use of an inappropriate kind of algorithm, and wrongly configured model parameter. This problem is often solved with a try and error approach which requires a trade of time and computational power. Gaining insight into the model and understanding the reasons why and when a model fails will improve the training process significantly.
- **Scientific understanding:** With growing datasets and complexity of problems, interpretability helps to find new understanding of real world causal relations within a model. When the data is too big to be monitored by humans, the trained model itself becomes a source of information.
- **Safety** is important in applications where wrong decisions of machines can have significant consequences (e.g. self driving cars). Because machine learning is mostly used for handling unknown data and uncertainty, it is hardly possible to test a model in all possible scenarios. To ensure the safety of a model, it is necessary to understand why it is taking a decision to find flaws in its reasoning.
- **Subconscious biases** become important in applications in which machines take decisions which directly affect humans. Some examples are loan approvals, job interview invitations or collision prevention in self driving cars. During the training process, machine learning algorithms can learn unwanted biases from the training data. This biases cannot be discovered by only evaluating the accuracy of a model, but via the impact of different features on a decision. Subconscious biases must be dealt with, to ensure fair decisions of machines.

- **Gaining trust:** There are fields of application for machine learning such as medical diagnosis or terrorism detection, where wrong decisions can be catastrophic. In these fields of application one cannot trust decisions on blind faith. In this context, trust is defined as the confident that the model will perform well on the real world objectives and scenarios, meaning that the user knows when the model perform well and when not [6]. If a human does not trust a machine learning model, he will and should not use it in critical applications.
- Since the **General Data Protection Regulation (GDPR)** which took effect in may 2018 it is not only useful to create interpretable models, but also required by law for many fields of applications. The GDPR regulates that in case of automated decision making, the data subject has the right to access ”meaningful information about the logic involved, as well as the significance and the envisaged consequences of such processing for the data subject”. [5, Articles 13-15 and Articles 21-22]

However, interpretability is not necessary for all machine learning models. If the consequences of an inaccurate result are not significant (eg. advertising) or the problem is well studied and validated in real application, there is no need to interpret the model. In this case the accuracy of the model is sufficient to evaluate the model's performance.

C. *Evaluating interpretability*

Methods to evaluate the interpretability of machine learning models vary depending on the paper or expert who is suggesting it. The evaluation of interpretability is especially difficult, because there is no consensus regarding an exact definition of machine learning interpretability.

A very general and therefore practical approach was suggested by Doshi-Velez and Kim [3]. Considering that good interpretation highly depends on the task and the user who receives the interpretation, a human grounded evaluation is suggested.

The most effective human grounded evaluation method is to evaluate the model with respect to its final task. For this purpose, experts of the domain evaluate the interpretability within a real application.
For example: If the task of a model is to work with doctors on the diagnosis of patients with a particular disease, the best way to evaluate could be performed by doctors performing diagnoses.
A good baseline for an experimental setup is to compare the model explanation with one of a human expert. This kind of evaluation aligns with methods used in human-computer interaction. On the down side, these methods are very expensive and must be well designed to reduce costs, time, and other resources.

To reach a compromise between efficiency and costs of evaluation, the task can be simplified, to the essence of the original task. For the evaluation of a simplified task, non-expert humans can be hired, allowing both a bigger subject pool and less expenses. [3]

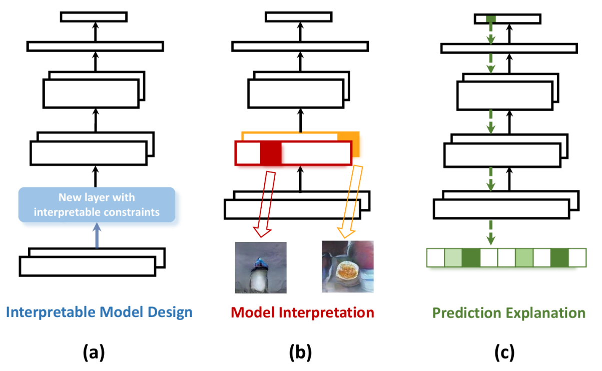

New layer with interpretable constraints
Interpretable Model Design
Model Interpretation
Prediction Explanation
(a)
(b)
(c)

Fig. 4. An illustration of three lines of interpretable machine learning techniques, taking DNN for example. (a) Interpretable model design. (b) Post-hoc interpretation of a model. (c) Post-hoc ex- planation of a prediction. [9]

IV. Methods to Interpret Machine Learning Models

This section explains different strategies to interpret machine learning models. The explanations and examples focus on deep neural networks and convolutional networks. However, the general approaches of interpretation can be applied to all machine learning models.

Depending on the time when an interpretation method is applied, it can be differentiated between designing interpretable models and post hock interpretability. This categories can further be differentiated, depending on the type of interpretation, into model level interpretation and prediction/instance level interpretation. [9]
The different interpretation methods are illustrated in figure 4 and explained in the following subsections. Sections V,VI and VII deepen the knowledge of specific interpretation methods for deep neural networks, which are most likely used in future research projects.

A. *Interpretable model design*

Model level interpretation can be achieved by designing models which incorporate interpretability directly in order to be globally interpretable or to explain their individual predictions.
There are two possibilities to construct globally interpretable models. Either the model is trained as usual but with interpretable constraints, or an interpretable model extraction is applied.

Early methods of interpretable constraints include the enforcement of sparsity which encourages a model to use less features for a prediction, or the imposing of semantic monotonic constraints to enable a monotonic relation between features and predictions. Hereby more comprehensibility for

the user is achieved, but the performance of the model is reduced significantly.
Modern methods try to incorporate semantically meaningful constraints to improve the model interpretability. An example method to interpret convolutional neural networks on model level is explained in detail in section V. [9]

Interpretable model extraction, or mimic learning, is the task to approximate a complex model using an easily interpretable model, such as decision trees, rule-based or linear models. Hereby the performance of the complex model is maintained and its statistical properties are reflected in the simple model. [9]

Prediction-level explainable models are used to give the user understandable rationale for a specific prediction. This can be achieved by applying an attention mechanism to the model, showing which part of the input is attended for a specific prediction. A representative example is the *Neural Image Caption Generation with Visual Attention* [11]. Here, a convolutional neural network is encoding an image to a vector representation which is used by a recurrent neural network with attention mechanism to generate an image description in text format.

B. Post-hoc interpretation

Post-hoc interpretation methods are used to extract information from pre-trained models. The concept can be applied without the need to sacrifice predicting performance and it provides useful information about knowledge stored in a model, or the reasoning behind a decision. The extracted information may be presented as a text explanation, visualization, local model explanation or by examples of other inputs. [6]
Post-hoc methods are categorized in model-agnostic methods which do not require model internal information and can therefore be applied to every kind of machine learning model and specialized methods which only work for one kind of machine learning model.

A representative model-agnostic interpretation method is the *Permutation Feature Importance* [13]. Its core idea is that some specific input features of a model are more important for a prediction than others. By permuting single input features, a contribution score can be calculated, depending on the permutations impact on the prediction.

Model-specific interpretation methods are often more powerful, because they do not ignore internal model information. For deep neural networks post-hoc interpretation methods are used to explain generally learned representations or specific prediction.
A very effective and widely used method to extract the representation learned by specific neurons in a deep neural network is to find the preferred input that maximized the neuron's activations. This method is called *activation maximization* and is explained in detail in section VI. [9]

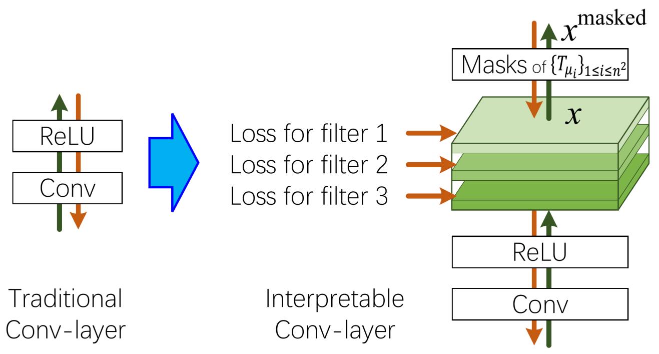


Fig. 5. Structures of an ordinary conv-layer and an interpretable conv-layer. Green and red lines indicate the forward and backward propagations. [10]

Using the hypothesis that larger gradients represent relevant features, back-propagation based interpretation methods are often used to explain predictions of a pre-trained neural network. Unfortunately, these methods are heuristically limited and may generate low quality explanations, which contain noise or highlight irrelevant features.
Guided feature inversion is a modern and very promising model specific interpretation method, which is used to explain predictions of convolutional neural networks. This method is explained in detail in section VII. [9]

V. Learn Disentagled Representations in Convolutional Neural Netsworks

In 2018 Zhang *et al.* [10] proposed a method to modify traditional convolutional neural networks in order to classify high level knowledge representations. For this purpose high level convolutional layers are replaced by interpretable convolutional layers.
Filters of a normal high level convolutional layer often represent a mixture of patterns, making the interpretability very difficult. Within an interpretable convolutional layer, every filter represents a specific object part, which is more semantically meaningful than representations of traditional convolutional layers. The difference between a traditional and an interpretable convolutional layer is illustrated in figure 8.

To train filters of interpretable convolutional layer to only represent one specific object, a loss function is added to the output of its feature map. This loss function encourages the filter to reduce the entropy of inter-category activations, as well as to reduce the entropy of spatial distribution of neural activations. This causes the filter to learn a distinct object part that belongs to a single object category and is only activate by one single part of the object, rather than on different object regions. [10]

VI. Activation Maximization

Activation Maximization is a model level, post-hoc interpretation method that is used to visualize representation learned by deep neural networks. It was first introduced by

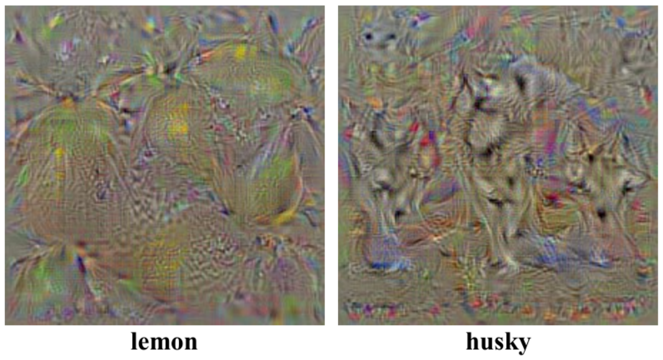


Fig. 6. Numerically computed images, illustrating the class appearance models, learned by a convolutional neural network, trained on ILSVRC-2013. [16]

Erhan et al. in 2009 [15] and further developed by many other researchers [16] [17] [18].

Activation Maximization is the task of finding an input pattern that maximally activates one or multiple target neurons in a specific layer of a neural network. The reasoning behind this is that the pattern could be a good first-order representation of what the neuron is doing. [15]

Activation Maximization is an optimization problem, mathematically described in equation 1 [9]:

$$x* = \underset{x}{\operatorname{argmax}} f_n^l(x) - \mathcal{R}(x) \tag{1}$$

Here, $x*$ is the input pattern that is being maximized, $f_n^l(x)$ the activation function at layer l for a specific neuron n and $\mathcal{R}(x)$ is a regulator. A local minimum for $x*$ can be found using back-propagation. This process is very similar to training a neural network, disregarding that the weights remain fixed, while the input is optimized. The regulator term in equation 1 is used to regulate low level image statistics, occurring in discriminatively-trained representations. Low level statistics are usually not interesting for a high level task. However, in practice, activation maximization is mostly used for image classification, where the maximized input images are used to understand the representations of classes and features. Low level statistics could result in spikes and other unwanted optical phenomenona which make the representation image look unrealistic [17].
Even though the framework is easy to use, getting it to work faces some challenges and surprises. Due to their large searching space, images which satisfy the optimization might still be unrecognizable (see figure 6). Different researchers approach these problems using proper regularization and image priors. [9]

Feature inversion, which was presented by Mahendran and Vedaldi in 2014 [17], is an approach to use a hand crafted image prior to optimize activation maximization. Instead of trying to maximize a single neuron, an example image is used

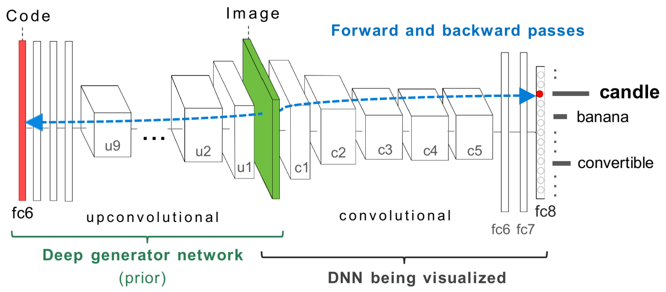


Fig. 7. Synthesizing a preferred input for a target neuron, using a deep generator network as image prior. In this example the deep generator network was trained to invert the feature representation of layer fc6. To find a preferred image, only the input of the deep generator network is optimized to maximized the activation of the target neuron. The gradient information is visualized as blue-dashed line. [18]

to compute the image encoding of a target layer (the encoding is represented by the activation matrix of that layer). Using only the layer representation, the original image is reconstructed. Starting from random values, the reconstructed image is optimized to closely match its target layer representation to the prior image representation.
This task is formally described by finding the image $x \in \mathbb{R}^{HxWxC}$ that minimizes the objective

$$x^* = \underset{x \in \mathbb{R}^{HxWxC}}{\operatorname{argmin}} ||f^l(x), f^l(x_0)||^2 - \lambda \mathcal{R}(x) \tag{2}$$

given a representation function $f^l(x) : \mathbb{R}^{HxWxC} \rightarrow \mathbb{R}^d$ at layer l and the original image x_0. To compare the image representation $f^l(x)$ with the original representation $f^l(x_0)$, the Euclidean distance is used. $\mathcal{R}(x) : \mathbb{R}^{HxWxC} \rightarrow \mathbb{R}^d$ is a regulator and λ balances the loss function and the regulator.
The core idea of feature inversion is that the image should not be completely invertible by its representation in intermediate layers, because deep neural networks ignore irrelevant information. Therefor, there should be a number of reconstructed images, showing an image interpretation $x*$ of the original image x_0 and all information that are used in intermediate layers to classify this particular image. [17]

In 2016 Nguyen et al. [18] improved the quality of activation maximization by using superior learned image priors, generated by a deep generator network. The image generator network was trained on the *image net* dataset to take in a vector of scalars and to put out a synthetic image that looks as real as possible.
The image generated by the generator network is used as input for a deep neural network. To activate a target neuron of the deep neural network, not the image itself but the input of the generator network is optimized. Figure 7 visualizes this setup. This method restricts the search to images that can be generated by the generator network, providing a strong bias to realistic images (depending on the quality of the generator network). [18]

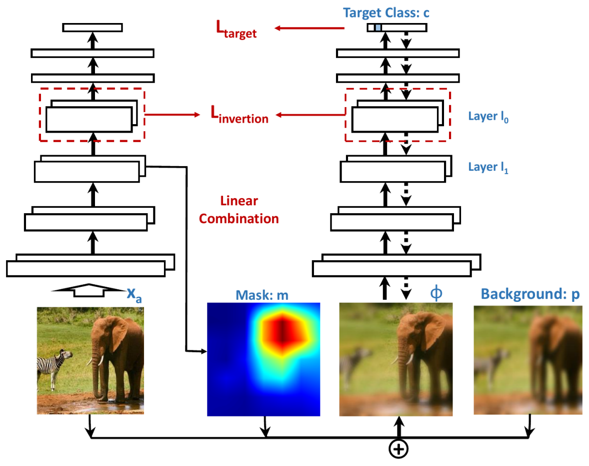


Fig. 8. Illustration of the different steps, necessary to perform a guided feature inversion. On the left, the layer representations are calculated for a specific image. On the right, the guided feature inversion and class-discriminative interpretation is applied to create a mask m, highlighting contributing features. This computational steps are explained in section VII. The linear combination of the layer l_1 is a regularization technique to improve the mask m, which is not explained of section VII but described in [14].

VII. Deep neural network based Prediction with Guided Feature Inversion

Guided feature inversion is a post-hoc interpretation method, taking advantage of intermediate layer information to explain predictions of convolutional neural networks. It is used in the field of image classification and computed in several steps, resulting in a mask that highlights the most contributing parts of a specific image. The computation steps are illustrated in figure 8.

In the first step the prediction for a target image x_0 is calculated, saving all representations $f^l(x_0)$ of the intermediate layers.
The feature matrix of the last convolutional layer l_0 is inverted using *guided feature inversion*. A guided feature inversion is an modified version of the feature inversion [17] (also referred to in section VI). In the guided feature inversion the expected inversion image x from equation 2 is reformulated as a weighted sum of the original image x_0 and a noise background image p.

$$\Phi(x_0, m) = x_0 \odot m + p \odot (1 - m) \tag{3}$$

The optimization target x from equation 2 is replaced with $\Phi(x_0, m)$ and a regulator is added to control the values of the weight matrix m.

$$L_{inversion} = ||f^{l_0}(\Phi(x_0, m)) - f^{l_0}(x_0)||^2 + \alpha \frac{1}{d} \sum_{i=1}^{d} m_i \tag{4}$$

The weight matrix $m \in [0,1]^d$ is optimized to minimize the inversion loss defined in equation 4. The first term corresponds to the inversion area, while the second term limits the area to be as small as possible. Without the second term, the error would be zero, if all entries within m equal 1. However if the parameter α, which balances the inversion error, and the area of m is set properly, m is optimized to highlighting the most contributing objects of the input x_0.

Because feature inversion is lacking discrimination power, the mask m might still highlight several class objects that do not belong to the target label. Using the last layer L activation, the results of the feature inversion are fine-tuned using a class-discriminative interpretation to only highlight objects of the target class c.
For this purpose m is inverted to a mask $m_{bg} = 1 - m$, highlighting irrelevant information with respect to the target class c. This includes the background and foreground objects of other classes. Using m_{bg}, the background part of the image can be calculated as weighted sum of the original image x_0 and p.

$$\Phi_{bg}(x_0, m_{bg}) = x_0 \odot m_{bg} + p \odot (1 - m_{bg}) \tag{5}$$

To remove contributing features that do not belong to the target class c from the mask, m is optimized to reduce the target loss, as defined in the following equation:

$$L_{target} = -f_c^L(\Phi(x_0, m)) + \lambda f_c^L(\Phi_{bg}(x_0, m)) + \beta \frac{1}{d} \sum_{i=1}^{d} m_i \tag{6}$$

The first term of equation 6 strongly activates the softmax probability f_c^L of class c while the second term is encouraging that no features that contribute to class c are located in the background. The last term is suppressing the area of m. The parameters λ and β are used to balance the different terms of the loss function.

More detailed explanations and further improvements of the interpretation method are described by Du *et al.* [14].

VIII. Conclusion

The research on machine learning interpretation methods has made significant process over the last few years. Especially for the interpretation of deep neural networks, many new algorithms were developed. Even though new interpretation methods seemingly improve in performance along with complexity, the insight into complex machine learning models is still limited. An especially challenging task is to evaluate the performance of different methods to compare them and to identify their practical usability. Up to this point the research of interpreting deep neural networks seems to remain solely academic. The question when the level of interpretation for deep neural networks is high enough to be used in practical applications, remains open.

References

[1] A. M. Turing (1950) Computing Machinery and Intelligence

[2] M. Awad and R. Khanna (2015) Efficient Learning Machines

[3] F. Doshi-Velez and B. Kim, (2017) Towards A Rigorous Science of Interpretable Machine Learning

[4] D. Gunning (2017) Explainable Artificial Intelligence (XAI) Program Update

[5] European Parliament (2016) General Data Protection Regulation

[6] Z. Lipton (2017) The Mythos of Model Interpretability

[7] I. Goodfellow and Y. Benigo and A. Courville (2016) Deep Learning

[8] K. Hornik (1989) Multilayer Feedforward Networks are Universal Approximators

[9] M. Du and N. Liu and X. Hu (2018) Techniques for Interpretable Machine Learning

[10] Q. Zhang and Y. Wu, and S. Zhu (2018) Interpretable Convolutional Neural Networks

[11] K. XU *et al.* (2016) Show, Attend and Tell: Neural Image Caption Generation with Visual Attention

[12] A. Nguyen and J. Yosinski and J. Clune (2016) Multifaceted Feature Visualization: Uncovering the Different Types of Features Learned By Each Neuron in Deep Neural Networks

[13] A. Altmann and L. Tolosi and O. Sander and T. Lengauer (2010) Permutation importance: a corrected feature importance measure

[14] M. Du and N. Liu and Q. Song and X. Hu (2018) Towards Explanation of DNN-based Prediction with Guided Feature Inversion

[15] D. Erhan and Y. Bengio and A. Courville and P. Vincent (2009) Visualizing Higher-Layer Features of a Deep Network

[16] K. Simonyan and A. Vedaldi and A. Zisserman (2014) Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps

[17] A. Mahendran and A. Vedaldi (2014) Understanding Deep Image Representations by Inverting Them

[18] A. Nguyen and A. Dosovitskiy and J. Yosinski and T. Brox and J. Clune (2016) Synthesizing the preferred inputs for neurons in neural networks via deep generator networks

[19] (23/02/2019) https://skymind.ai/wiki/neural-network

[20] R. Batuwita (2016) Deep Learning: Towards General Artificial Intelligence