TU Delft student report

A SURVEY ON CONVOLUTIONAL NEURAL NETWORK EXPLAINABILITY METHODS

Nikki Bouman
4597648
N.H.Bouman@student.tudelft.nl

Vanisha Jaggi
4576926
V.A.Jaggi@student.tudelft.nl

Mostafa Khattat
4588231
M.Khattat@student.tudelft.nl

Nima Salami
4589238
N.Salami@student.tudelft.nl

Victor Wernet
4554582
V.G.A.Wernet@student.tudelft.nl

Wouter Zonneveld
4582861
W.R.Zonneveld@student.tudelft.nl

Department of Computer Science
Delft University of Technology

October 30, 2019

ABSTRACT

Artificial Intelligence (AI) is increasingly affecting people's lives. AI is even employed in fields where human lives depend on the AI's decisions. However, these algorithms lack transparency, i.e. it is unclear how they determine the outcome. If, for instance, the AI's purpose is to classify an image, the AI will learn this from examples provided to it (e.g. an image of a cow in a meadow). The algorithm can focus on the wrong part of the image. Instead of focusing on the foreground (cow), it could focus on the background (meadow). This way, by focusing on the background, it could produce a false output (e.g. a horse instead of a cow). To show this, an explanation is needed. For this reason, a variety of methods have been created to explain the reasoning behind these algorithms, called explainability methods. In this paper, six local explainability methods are discussed and compared. These methods were chosen as they were the most prominently used approaches for explainability methods for Convolutional Neural Networks (CNN). By comparing methods with analogous characteristics, this paper is going to show what methods exceed others in terms of performance. Furthermore, their advantages and limitations are being discussed. The comparison shows that Local Interpretable Model-agnostic Explanations, Layer-wise Relevance Propagation and Gradient-weighted Class Activation Mapping perform better than Sensitivity Analysis, Deep Taylor Decomposition and Deconvolutional Network, respectively.

Keywords Explainability · AI · Interpretability · CNN · Convolutional Neural Networks · Sensitivity Analysis · LIME · Deep Taylor Decomposition · LRP · DeconvNet · Grad-CAM

1 Introduction

Artificial Intelligence (AI) is an ever growing field of Computer Science and is widely applied in fields ranging from speech recognition [27] to recruiting tools [9]. Although for some tasks AI systems have achieved accuracies on par with humans [10], their decisions can lead to life or death situations when applied in military [14] or medical fields [22]. Since AI's decisions influence people's life more and more, ethical concerns and a need for an explanation on how these algorithms make their decisions arise. Furthermore with the introduction of General Data Protection Regulation (GDPR), people in the EU have the right to "obtain an explanation of the decision" made by an AI system [29]. Due to deep learning model's nested non-linear structure, it is unclear what information the input data makes them arrive at their decision. Today's AI systems are trained with many examples, which may cause them to observe patterns in the data which are not directly visible to the person analyzing the data. This can make it impossible for humans to understand their reasoning [2]. A new field has emerged which attempts to explain these AI's called Explainable Artificial Intelligence (XAI). By using XAI people can try to extract explanations of AI systems and gain new insights [32] in deep neural networks.

An artificial neural network (ANN) is a branch of AI which is used to recognise patterns in data. An ANN consists of many neurons which are grouped together in layers as shown in Figure 1. Neurons have an activation function and a bias. Neurons are connected to each other via edges which have weights. When an input x (e.g. an image) is given to the ANN, it first enters the so called *input layer*. The signal is now propagated to the next layer using the edges' weights and the neurons' activation functions and biases to compute an output which in turn is propagated to the next layer. This process continues until the final layer, the so called *output layer*, is reached. The output layer now contains the prediction $f(x)$ (the probability of the input containing the target class) of the ANN. This prediction can be made for a single class c, but typically this prediction is made for multiple classes [40].

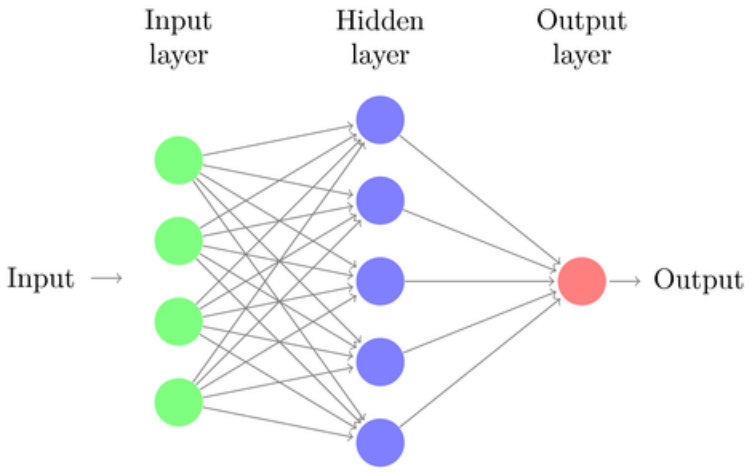

Figure 1: Overview of the layout of a simple neural network. Adapted from [12].

All layers between the input and output layer are called *hidden layers*. When an ANN consists of multiple hidden layers, it is called a deep neural network (DNN). A convolutional neural network (CNN), is a class of DNN, designed for image analysis. The distinctive property of CNNs is that a CNN uses the mathematical operation *convolution* in its layers. During the convolution operation, filters are used to extract features which are relevant for certain classes. Nearly all CNNs use *pooling* which summarizes statistics of nearby outputs, reducing the size and complexity of the neural network [15].

This paper discusses several explainability methods which are used to explain CNNs. These methods are either model-agnostic or model-specific. The latter can only be applied to specific model classes, while model-agnostic methods can be applied to any machine learning algorithm [2]. Furthermore, explainability methods can be either global or local. Local interpretations mean that the results of a trained model on a specific input can be understood, while global interpretability is about understanding the entirety of a trained model [13]. This paper only discusses local methods.

The question this paper will answer is: what frequently used explainability methods exist for CNNs? In addition, the advantages and limitations of these methods will be explored and compared. Finally, the explainability methods will be distinguished into model-specific and model-agnostic.

In section two of this paper, six explainability methods are described. Features of each method are discussed, followed by their advantages and limitations. By knowing the advantages and the limitations, this paper will compare the six different explainability methods in section four. At the end a discussion and a conclusion are provided.

2 Explainability Methods

When using a CNN to search for some class c (e.g. a dog/cat) in an image, a frequently used way to explain the CNN's prediction $f_c(x)$, is to highlight areas of the input which significantly influenced the CNN's prediction. This can be done by using a heatmap (see Figure 2). The heatmap is built up from relevance scores R for each pixel i. A high R_i^c means that the pixel contributed significantly to identifying the class c. Typically areas with high relevance scores are located at key characteristics of c. For example, in Figure 9 the CNN has determined that the face of the dog is an important feature of the image. This section will give a short description of how different explainability methods construct a heatmap.

2.1 Sensitivity Analysis (SA)

One classic method for explaining predictions is Sensitivity Analysis (SA) [23, 32]. SA explains a prediction based on the model's locally evaluated gradient. The relevance score R is defined as:

$$R_i^c(x) = \| \frac{\delta}{\delta x_i} f_c(x) \| \tag{1}$$

This measure assumes that the most relevant input features are those to which the output is most sensitive. The result of the explanation process is a heatmap, which indicates which pixels need to be changed to make the image look more/less like the predicted class.

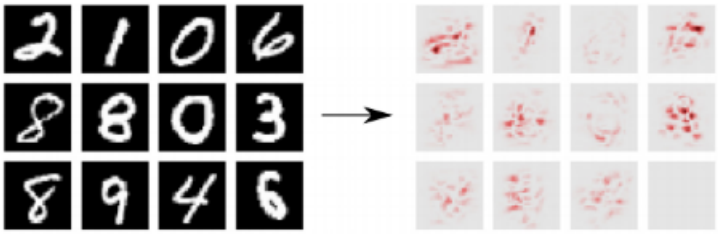

Figure 2: Sensitivity Analysis applied to a convolutional deep neural network trained on handwritten digit database, and the resulting explanations (heatmaps) for selected digits. Red color indicates positive relevance scores. Taken from [23].

Examples of explainability methods produced by SA is given in Figure 2. Each heatmap has a scattered outcome and does not focus on the actual class-relevant features. This can be blamed on the nature of SA: it is not only sensitive to pixels of the object of the class, but it could also be sensitive to objects of other classes. The heatmap indicates what pixels make the digit belong more/less to the target class rather than which pixels are actually pivotal for that class [23, 2].

Performing SA can have different purposes. One purpose for using SA is to use it as a tool to find and remove unimportant input attributes. SA is mostly used as a starting point for some more powerful explainability method [2]. Five more methods that are known as an improvement on SA will be discussed next.

2.2 Local Interpretable Model-agnostic Explanations (LIME)

Local Interpretable Model-agnostic Explanations (LIME) [30], is a model-agnostic method which produces explanations that are simple and clear so that it is easy for a human to understand the underlying functioning of the models at hand. LIME approximates the black box locally in the neighborhood of the prediction being explained. The top three ways LIME showed in experiments carried out by Ribeiro et al. [30] to have helped users are: choosing between competing models, detecting and improving untrustworthy models and lastly, getting insights into the models [30].

$$\xi(x) = \underset{g \in G}{\operatorname{argmin}} \mathcal{L}\left(f, g, \pi_x\right) + \Omega(g) \tag{2}$$

The explanation of LIME is produced by optimizing Equation 2. This explanation is defined as a model $g \in G$ with G being a class of potentially interpretable models and g is either 0 or 1 which means g shows whether or not an

interpretable component exist and $\Omega(g)$ is the measure of complexity (as opposed to interpretability) of the explanation of that model. In the model that is being explained, $\pi_x(z)$ is a proximity measure between an instance z to x to define the locality around x. Lastly, fidelity function $\mathcal{L}(f, g, \pi_x)$ is to measure how unfaithful g is in approximating f in the locality defined by π_x. Fidelity in this mathematical context refers to "the faithfulness of technology-based behavior and properties of virtual objects to the mathematical behavior and properties of the objects they are intended to represent" [26]. To make sure both interpretability and local fidelity are ensured, we should aim to minimize $\mathcal{L}(f, g, \pi_x)$ while trying to have $\Omega(g)$ as low as possible enough to be still interpretable.

Since in Equation 2 the formulation can be used with different explanation and interpretability families of G and different fidelity functions $\mathcal{L}$ or complexity measures of Ω, LIME can be used for any sort of classifier or regressor, and image processing is one of the cases where it is used the most in the field nowadays [19].

(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

Figure 3: Explaining an image classification prediction made by Google's Inception Neural Network. The top three classes predicted are "Electric Guitar" (output q = 0.32), "Acoustic guitar" (p = 0.24) and "Labrador" (output q = 0.21). Taken from [30].

One example of the applications of LIME is explaining the prediction of Google's pre-trained Inception Neural Network [38] on an arbitrary but absurd photo of a Labrador playing guitar (Figure 3a). In 3b, 3c and 3d we can see the explanations that are given for the top three predicted classes. Such explanations help humans in increasing trust in the classifiers.

Since the introduction of LIME in 2016 [39], different variations of LIME have been introduced for various use cases. To recap, the original version is an algorithm that can explain the predictions of any classifier or regressor in a faithful way, by approximating it locally with an interpretable model. Besides that, some of the other most commonly used variations are: • SP-LIME, a method that selects a set of representative and non-redundant predictions with explanations to address the problem of "trusting the model", via sub-modular optimization [30]. • LIME-SUP, a locally interpretable model based on supervised partitioning of input based on fitting trees to the fitted response [17]. • KLIME, an approach in contrast with LIME-SUP that is based on clustering the predictor space [17].

2.3 Deep Taylor Decomposition (DTD)

The Deep Taylor Decomposition (DTD) [6] is a method that explains a model's decision, specifically a DNN by decomposing the function value $f(x)$ as a sum of relevance scores [23]. The relevance redistribution onto a neuron becomes the following equation [25]:

$$R_i^n(x) = \sum_j \frac{w_{ik}^2}{\sum_{i'} w_{i'k}^2} R_k^{n+1}(x) \tag{3}$$

In this formula, w is defined as the weights connected between node i and node k, n is an index for some layer in the neural network, and $R_k^{n+1}(x) > 0$. Such that its nearest root point is the intersection of the plane equation and the line of maximum descent. Equation 3 is also known more commonly as the "w^2-rule" [24]. The propagation rule simply consists of redistributing the relevance score based on the square magnitude of the weight, and pooling relevance across all neurons j [25]. This relevance score approach was described by Bazen and Joutard [6] as a non-linear generalization of the Oaxaca method in econometrics [28]. A visual representation can be seen for the search direction where the gradient is the largest in Figure 4.

Knowing this, it is now possible to find the dominant (a class with the maximum output that describes the image), and non-dominant classes in an image [31]. As an example, a prediction for the class "cat" is obtained by forward-propagation of the pixel values x_i, and is encoded in $f(x)$. The output neuron is assigned a relevance score $R_i(x) = f(x)$ representing the total evidence for the class "cat". Relevance is then backpropagated from the top layer (layers closer to

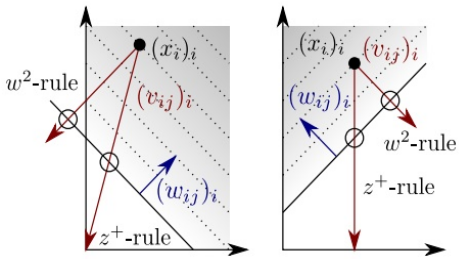

Figure 4: Illustration of a root point search in the two-dimensional input space. The data point $(x_i)_i$ is represented as a black dot, and the possible root points are depicted as circles. Taken from [24].

the output) down to the input, where R_i denotes the pixel-wise relevance score, that can be visualized as a heatmap [25]. It is also easier in the lower layers where the relevance score has been redistributed to the relevant neurons, and where the final redistribution step only pertains to adding the neighboring pixels together [25]. A visual example can be seen in Figure 5.

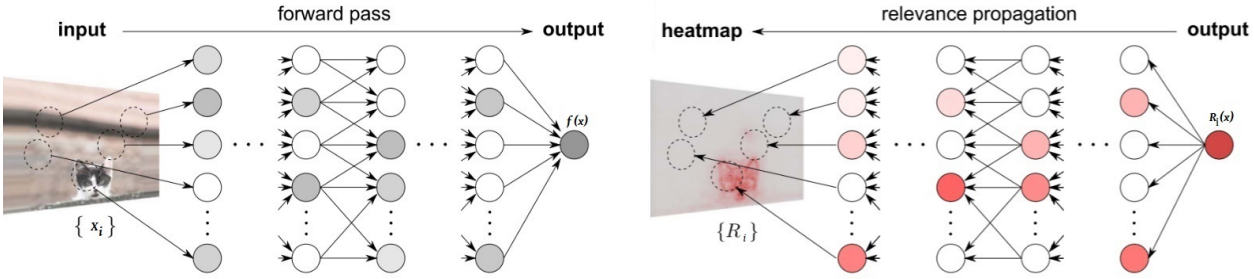

Figure 5: On the left: the forward pass. On the right: the relevance propagation. Adapted from [25].

2.4 Layer-wise Relevance Propagation (LRP)

Layer-wise relevance propagation (LRP) was first introduced by Bach et al. [5] as a concept defined by a set of constraints. There is one main constraint from which more constraints can be derived. This constraint is that the relevance conservation property (no relevance score is created or lost) is ensured, thus:

$$\sum_{i=1}^{V} R_i^0 = ... = \sum_j R_j^n = \sum_k R_k^{n+1} ... = f(x) \tag{4}$$

Here V is defined as the size of the input. This implies that the prediction $f(x)$ can be decomposed into a sum of the relevance scores of all the input pixels. The relevance scores are then visualized in a heatmap.

LRP uses backpropagation to calculate its result. This means that LRP starts from the output layer and layer-wise calculates how the previous layer influenced the next layer. How exactly the relevance is propagated is determined by the propagation rule. Several approaches are possible as long as the aforementioned constraint is preserved. As the $\alpha\beta$-rule has been shown to work well, this paper will focus on this particular propagation rule [23]:

$$R_j^n = \sum_k (\alpha \frac{a_j w_{jk}^+}{\sum_j a_j w_{jk}^+} - \beta \frac{a_j w_{jk}^-}{\sum_j a_j w_{jk}^-}) R_k^{n+1} \tag{5}$$

In this formula, a_j is the activation function of neuron j in layer n, w_{jk}^+ and w_{jk}^- are the positive and negative weights respectively of the edges in between neurons j and k which are in layers n and $n+1$, α and β are parameters which can be chosen. An interesting property of LRP is thus that there is not only positive relevance, but also negative relevance. α and β, must be chosen such that: $\alpha - \beta = 1$ and $\beta \geq 0$.

When setting $\alpha = 1$ and $\beta = 0$ (also denoted as LRP-$\alpha_1\beta_0$) an approximation of the Deep Taylor Decomposition is obtained, meaning there is no negative relevance. Increasing α and β results in more negative relevance in the final heatmap. What value of α should be used depends on the application, i.e. on GoogleNet LRP-$\alpha_1\beta_0$ is stable while for image recognition LRP-$\alpha_2\beta_1$ works well [23, 25].

2.5 Deconvolutional Network (DeconvNet)

The Deconvolutional Network (DeconvNet) method [1, 16] was proposed for unsupervised feature learning, however, it is currently applied to visualization. This means that similar to the other mentioned methods, it reveals which pixels of the input image are responsible for its visualization. The more one pixel contributes to the visualization, the more important that specific pixel becomes. This computation is done by using backpropagation.

The importance of pixels can be shown by using the prediction of classes, the class score, which originally is a highly non-linear function. Since the non-linear function is too difficult to work with, it is simplified to a linear function by using the first-order Taylor expansion, the Linear Class Score:

$$f_c(x) = R^c x + b_c \tag{6}$$

where b is the bias of the model and R^c the heatmap. Pixels which have to change the least (change of pixels can be computed with Equation 1) to change the class score model (Equation 6) the most are the more important pixels [16, 37].

DeconvNet also constructs a Class Activation Map (CAM). This reveals the discriminative image regions which are relevant to a particular category identified by the CNN [7, 16]. CAM requires a special network architecture (see Figure 6) similar to Network in Network [21] and GoogLeNet [38] in which the network consists of convolutional layers and the network performs a global average pooling on the convolutional features before the final output layer (e.g. softmax layer in the case of classification).

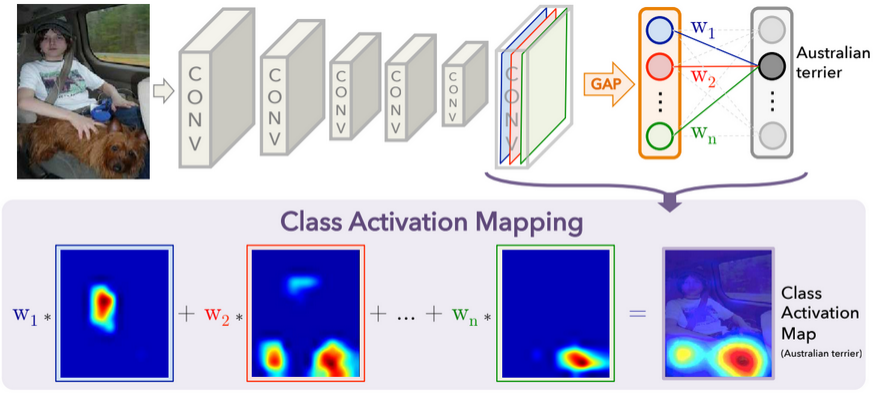

Figure 6: Class Activation Mapping highlights the regions related to a particular class. Taken from [7].

For a given image x, let A^k_{ij} represent the feature map of unit k in the last convolutional layer at spatial location (i, j). For each unit k, the result of global avarage pooling is $P^k = \sum_i \sum_j A^k_{ij}$. The final class score f_c can be written as:

$$f_c = \sum_k w^c_k P^k \tag{7}$$

In Equation 7 w^c_k indicates the importance of class c for unit k by training a linear classifier for each class c using activation maps of the last convolutional layer generated for the given image x [8]. By plugging P^k into the class score f_c the heatmap can be obtained [7]:

$$R^c_{CAM} = \sum_k w^c_k A^k_{ij} \tag{8}$$

This class activation heatmap R^c_{CAM} is shown for the example class 'dog' at the bottom row of Figure 6.

2.6 Gradient-weighted Class Activation Mapping (Grad-CAM)

Gradient-weighted Class Activation Mapping (Grad-CAM) [34] is a strict generalization of CAM. CAM is limited to a special network architecture and requires to retrain the network. Grad-CAM has been proposed to address these issues. It is applicable to any CNN-based model without architectural change or retraining.

In Grad-CAM the weights w^c_k can be obtained without retraining the network by first computing the gradient of class score f_c, or any other differential activation, with respect to the feature maps A. Then a global average pool of these

gradients will be taken and normalized using the number of pixels in the activation map Z, as a measure of scale [35]:

$$w_k^c = \overbrace{\frac{1}{Z}\sum_i\sum_j}^{\text{global average pooling}} \underbrace{\frac{\delta f_c}{\delta A_{ij}^k}}_{\text{gradients via backprop}} \tag{9}$$

As in Equation 8 the Grad-CAM heatmap is a weighted combination of feature maps:

$$R^c_{Grad-CAM} = ReLU \underbrace{\left(\sum_k w_k^c A^k\right)}_{\text{linear combination}} \tag{10}$$

where the ReLU (Rectified Linear Unit) activation function is applied to select features with a positive influence on the class of interest as Grad-CAM may have negative values that likely belong to other categories in the image rather than the class of interest [34].

Grad-CAM obtains class-discriminative and localization map by producing only a coarse heatmap which does not give a detailed explanation of why a network predicts a particular class of interest. For example if the classifier predicts a cat in the image as a tiger cat, the heatmap does not give a clear visualization of important features such as stripes, pointy ears and eyes [34] (Figure 7).

Figure 7: Grad-CAM visualization is class-discriminative but lacks details of visualization. Adapted from [34].

To work around this issue Grad-CAM is fused with an existing high-resolution visualization method namely Guided Backpropagation which produces the fine-grained importance of an image (Figure 8). This new approach of visualization, called Guided Grad-CAM, performs a point-wise multiplication between the scores obtained from Grad-CAM and the scores obtained from Guided Backpropagation [34] (Figure 9).

Figure 8: Guided Backpropagation is unable to distinguish the features of dog and cat but can show a high-resolution visualization. Adapted from [34].

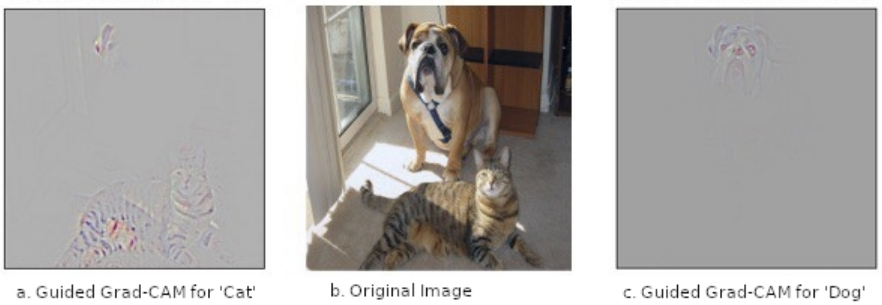

Figure 9: The result of combining the best aspects of Guided Backpropagation and Grad-CAM into a method called Guided Grad-CAM. Adapted from [34].

The result will be a high-resolution class-discriminative visualization [34] where the classes 'cat' and 'dog' are now clearly discriminated.

3 Advantages and Limitations

In this section the advantages and limitations of each method mentioned in this paper will be discussed. Before delving into the details, it should be noted that there is one common advantage between all methods, namely visualization. Visualization helps to identify dataset bias and lends insight into failures of CNNs. This is why it is important to have fair and ethical outcomes in real world applications [34].

3.1 Sensitivity Analysis (SA)

SA is a classical method and is commonly used as a starting point for some more powerful explanation method, or as a tool to find and remove unimportant attributes [2]. However, SA does not produce an explanation of the function value itself. The produced heatmaps therefore do not focus on the actual class-relevant features [2, 23].

3.2 Local Interpretable Model-agnostic Explanations (LIME)

The strongest points of LIME fall in two folds, local interpretability and human readability. When applying LIME, the final output is a list of explanations which reflects the contribution of *each* interpretable feature to what is predicted on the data sample, this helps to interpret the output locally. Additionally, to make the results more readable for humans, the way LIME explains the results is in line with what people are interested in when looking at the output of a model. It modifies a single sample data by making changes to the feature values and then looking out for the effects that this specific feature has on the final output [2, 19].

However, LIME falls short in two areas, namely, linear models exclusivity and use case specificity. The original LIME implementation only focuses on linear models that can approximate the local behaviour. Especially, for the case of complex datasets, non-linearity is a likely case for local regions and LIME is not able to give good explanations for such cases [18]. Moreover, the tweaks and modifications that are applied on the datasets in order to obtain the explanations needed, are vastly specifically use case based. That means in many situations, simple perturbations are not enough while in the best case scenario, these perturbations should be in effect through the differences that are seen in the dataset. But also, intentionally and manually tweaking the perturbations is not ideal either, since it could potentially add bias to the model explanations [19].

3.3 Deep Taylor Decomposition (DTD)

DTD efficiently utilizes the structure of the network by backpropagating the explanations from the output to the input layer [31]. When applying this method to classes other than the dominant class, the explanation does not precisely describe the features of the non-dominant class(es). This effect is known as diffusion, as the explanation for non-dominant classes is diffused by the explanation for the dominant class. For this situation, DTD tends to fail drastically when trying to explain multiple classes in a single image.

3.4 Layer-wise Relevance Propagation (LRP)

The first advantage of LRP is that it uses a continuous function, which means that nearly identical inputs will output nearly identical explanations [23]. Furthermore LRP has a high selectivity, meaning that the average value of $f(x)$ decreases steeply when the most relevant patches of pixels are removed. This in turn means that it is good at selecting what part of the input is relevant for classification [23]. Finally LRP is applicable on any neural network with monotonous activations [33].

However, since LRP is using layer-wise backpropagation, it is possible for some of the relevance score to reach a "dead-end" in the lower layers which results in the relevance score being distributed randomly [5].

3.5 Deconvolutional network (DeconvNet)

DeconvNet is designed to find which visual patterns cause deep neurons to fire strongly. This should characterize the selectivity of neurons. However, it fails to produce selective signals from deep neurons, which as a consequence, generates a rather uniform response. The uniform response shows that DeconvNet is not selective for foreground information. What is does well is that it accurately reproduces the object shapes as well as the image boundaries for visualization of images [1]. However, it seems to perform partial input recovery, which is a consequence of DeconvNet being invariant to network reparamaterizations under certain conditions [1, 3].

3.6 Gradient-weighted Class Activation Mapping (Grad-CAM)

The main advantage of Grad-CAM is that it can be applied to the existing CNN classifications without requiring architectural changes. Another strong point of Grad-CAM is being robust against adversarial noises. In an experiment done by Selvaraju et al. [34] the network was fooled into thinking that the result of an image classifier belongs to a wrong class with high confidence but in spite of that the Grad-CAM could still localize the original categories successfully [34].

However, there are also drawbacks. In the case of an image with multiple occurrences of the same class, Grad-CAM fails to properly localize all of them in the image. In addition, Grad-CAM often fails to localize the entire object. As a result, only a part of the object is visible in the visualization [8].

4 Comparison

As SA and LIME are the only model-agnostic methods discussed in this paper, they will be compared. The other four methods are considered model-specific. DTD and LRP are methods which can both be applied to DNNs and CNNs, see Table 1, while DeconvNet and Grad-CAM can only be applied to CNNs. Therefore, these methods will be pair-wise compared.

	Any model	**DNN**	**CNN**
SA	✓	✓	✓
LIME	✓	✓	✓
DTD		✓	✓
LRP		✓	✓
DeconvNet			✓
Grad-CAM			✓

Table 1: Overview of the applicability of the explainability methods.

4.1 SA vs. LIME

When looking at Table 1, it is quite easy to make the mistake of believing that SA and LIME are similar methods. Although both methods are designed to be applicable to any machine learning model [23, 30], in the way each algorithm works there are fundamental differences. SA is one of the first and simplest approaches attempting to explain the results of machine learning models [32]. This level of simplicity is a downfall of SA as it is one of the least accurate and least efficient models among those that have been covered in this paper [2]. LIME however was designed relatively recent and with the goal of addressing shortcomings of SA in mind [17]. The designers of LIME took a different approach and came up with a more robust way of explaining machine learning models. Currently LIME and its latest variations are among the most popular explainability methods in this field [11, 19].

4.2 DTD vs. LRP

DTD is known to be motivated by a linear Taylor decomposition for every individual neuron and excitation back-propagation by a probabilisitic "Winner-Take-All" approach [4]. When working with LRP it is crucial to use various propagation rules for every single layer, since it has been shown to be empirically useful to apply different rules on different parts of the network [4]. DTD requires different propagation rules on the input data range [25]. LRP has been shown to be stable and was able to quickly explain the predictions on a broad range of classifiers [20].

4.3 DeconvNet vs. Grad-CAM

One of the approaches of DeconvNet is CAM [7]. Grad-CAM and CAM can only be applied to CNNs. CAM however requires a special CNN architecture with a global average pooling layer after the last convolutional layer and before the final output layer [34] (see Figure 6). It also requires retraining of multiple linear classifiers (one for each class) [8]. The change of architecture in tasks like image classification limits the network model and may results in inferior accuracies [34]. On the other hand Grad-CAM does not suffer from those limitations and can be applied to any CNN.

4.4 Performance graphs

Determining objectively whether an explainability method works well or not, can be difficult. A systematical approach to compare heatmaps would give a quantitative measure of which explainability method is better. A method to test an explainability method's selectivity is proposed by Samek et al. [32] and Bach et al. [5] and is called *pixel flipping*. The pixel flipping method removes patches of 4x4 pixels starting with the patch of pixels with the highest relevance score. If the average function $f(x)$ drops steeply, the explainability method has a high selectivity for relevant areas in the image.

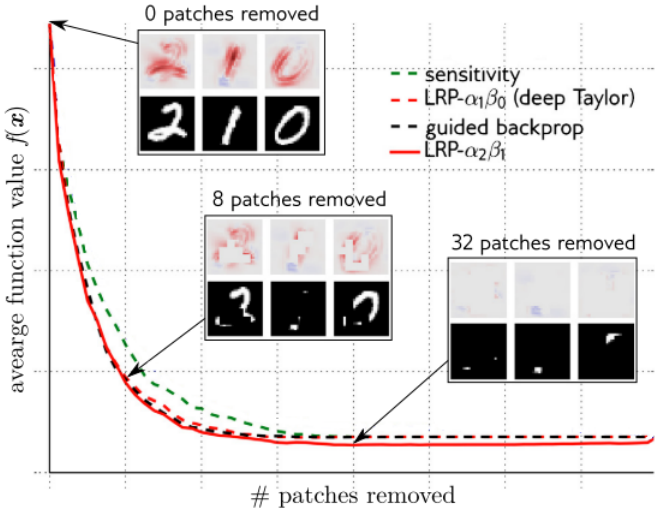

Figure 10: "Illustration of the "pixel-flipping" procedure. At each step, the next most relevant region according to the heatmap is removed (by setting it to black), and the function value $f(x)$ is recorded". Taken from [23].

Figure 10 shows for different explainability methods how $f(x)$ drops as patches of pixels are removed. LRP-$\alpha_2\beta_1$ has the steepest drop in average $f(x)$ value, closely followed by Guided Backpropagation and LRP-$\alpha_1\beta_0$ (DTD). The SA method is performing rather poorly.

Another measure to compare explainability methods proposed by Samek et al. [32] is the size of the produced heatmaps. A heatmap which contains more complexity (i.e. sparsity or randomness) cannot be compressed as much as heatmaps which contain only relevant regions, thus increasing their size. An explainability method which produces smaller sized heatmaps focuses better on important parts of the input.

In Figure 11 it is clear that LRP has the lowest average entropy in images from three separate image data-sets (SUN397, ILSVRC2012, and MIT Places). This results in a lower heatmap size for LRP than both SA and DeconvNet. Therefore LRP is better in identifying relevant structures in an image.

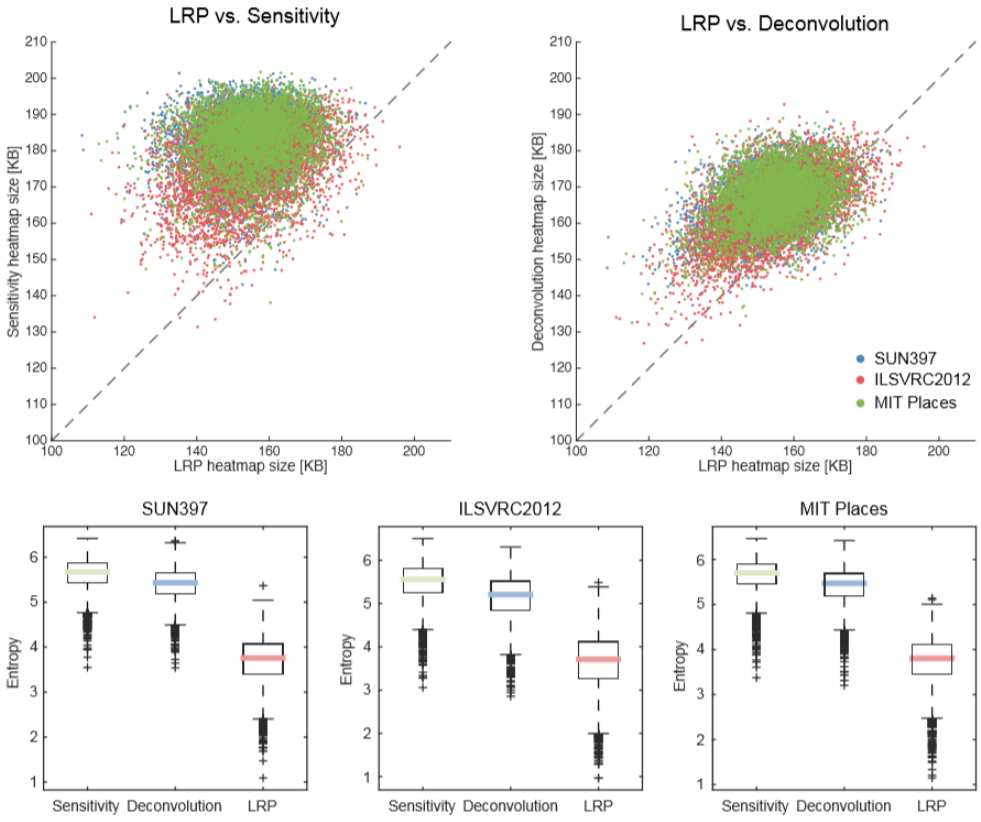

Figure 11: "Comparison of heatmap complexity, measured in terms of file size (top) and image entropy (bottom)". A smaller heatmap means better focus on the important features of the input. Taken from [33].

5 Discussion

This paper explains, evaluates, and compares six explainability methods for CNNs. These methods were chosen as they are most prominent when searching for explainability methods for CNNs. However, other methods for explaining CNNs exist, e.g. DeepLIFT [36]. Moreover, some of the discussed methods have different variations. For example, Grad-CAM++ is a generalization of Grad-CAM and addresses the issues and limitations of Grad-CAM [8]. Also LIME has many variations such as SP-LIME, LIME-SUP, and KLIME [17, 30].

Furthermore, this paper has built up a comparison between various methods to illustrate and express their differences. A challenge with explainability methods is to measure and compare their performance. Quantifiable tests such as pixel flipping [5, 32] and size comparisons [32] have been been created to objectively compare different explainability methods. However, comparative data is frequently not available since the quality of an explainability method is often assessed by manual examination of the produced heatmap.

This discussion provides motives to do more research on explainability methods for CNNs. Further research is necessary to provide a more comprehensive overview of contemporary explainability methods for CNNs. This overview should contain the advantages and limitations of each method to easily identify which method is favorable in different use cases.

6 Conclusion

AIs make decisions which impact people's lives significantly. Their reasoning is often unclear, which raises a need for methods to explain AIs decisions. A frequently used method to explain AIs is visualization. A good visualization technique is a heatmap which displays which areas of the input significantly influenced the network's prediction. Six frequently used explainability methods for CNNs are reviewed in this paper.

Sensitivity Analysis (SA) is a classic and relatively older explainability method which is outperformed by the other methods [2]. Local Interpretable Model-agnostic Explanations (LIME) gives simple and human readable explanations on how each interpretable feature is contributing to the final regressor results. This helps in interpreting the output locally [11, 17, 30]. SA and LIME are model-agnostic whereas the other four are model-specific. From these two methods, LIME is often chosen over SA as it provides a clearer explanation of the classification.

Deep Taylor Decomposition (DTD) efficiently utilizes the structure of the network by backpropagating the explanations from the output to the input layer [31]. Layer-wise Relevance Propagation (LRP) is a backpropagation method which allows for both positive and negative relevance. LRP outperforms both SA and DeconvNet in complexity and sensitivity [23, 33]. DTD and LRP are similar methods, however LRP has the advantage of supporting negative relevance which makes it perform better in certain models.

Deconvolutional network (DeconvNet) looks at the pixels of an image that need to change the least to affect the classification prediction of a class the most. Besides, it constructs a Convolutional Activation Map [16, 37]. Gradient-weighted Class Activation Mapping (Grad-CAM) is a generalization of the CAM method combined with Guided Backpropagation which works with any CNN model and produces a high-resolution visualization [34]. Grad-CAM is an improved version of CAM which is used in DeconvNet. Futhermore, Grad-CAM is applicable on a wider variety of networks.

This paper only compares six explainability methods, while in practice, more methods exist. Hence for future work, a more exhaustive comparison would help in finding which method performs best. Moreover, having more quantifiable comparisons of the different methods is necessary to further explore their performance.

References

[1] Mahendran A. and Vedaldi A. "Salient deconvolutional networks". In: *Lecture Notes in Computer Science (eds) Computer Vision - ECCV 2016*. Ed. by Welling M. ECCV 2016. Leibe B. Matas J. Sebe N. Vol. 9910. Springer, Cham, 2016.

[2] Amina Adadi and Mohammed Berrada. "Peeking inside the black-box: A survey on Explainable Artificial Intelligence (XAI)". In: *IEEE Access* 6 (2018), pp. 52138–52160.

[3] Julius Adebayo et al. "Sanity Checks for Saliency Maps". In: *Advances in Neural Information Processing Systems 31*. Ed. by S. Bengio et al. Curran Associates, Inc., 2018, pp. 9505–9515. URL: http://papers.nips.cc/paper/8160-sanity-checks-for-saliency-maps.pdf.

[4] Maximilian Alber. "Software and application patterns for explanation methods". In: (2019). arXiv: 1904.04734.

[5] Sebastian Bach et al. "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation". In: *PloS one* 10.7 (2015), e0130140.

[6] Stephen Bazen, Xavier Joutard, et al. *The Taylor Decomposition: A Unified Generalization of the Oaxaca Method to Nonlinear Models*. Tech. rep. Aix-Marseille School of Economics, Marseille, France, 2013.

[7] Zhou Bolei et al. "Learning Deep Features for Discriminative Localization". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2921–2929.

[8] Aditya Chattopadhyay et al. "Grad-CAM++: Generalized Gradient-based Visual Explanations for Deep Convolutional Networks". In: *IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2018, pp. 839–847. URL: https://ieeexplore.ieee.org/document/8354201.

[9] Jeffrey Dastin. *Amazon scraps secret AI recruiting tool that showed bias against women*. Oct. 2018. URL: https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scraps-secret-ai-recruiting-tool-that-showed-bias-against-women-idUSKCN1MK08G (visited on 10/01/2019).

[10] Nicola Davis. *AI equal with human experts in medical diagnosis, study finds*. Sept. 2019. URL: https://www.theguardian.com/technology/2019/sep/24/ai-equal-with-human-experts-in-medical-diagnosis-study-finds (visited on 10/01/2019).

[11] Vincenzo Di Cicco et al. "Interpreting Deep Learning Models for Entity Resolution: An Experience Report Using LIME". In: *Proceedings of the Second International Workshop on Exploiting Artificial Intelligence Techniques for Data Management*. aiDM '19. Amsterdam Netherlands: Association for Computing Machinery, 2019. ISBN: 9781450368025. DOI: 10.1145/3329859.3329878. URL: https://doi.org/10.1145/3329859.3329878.

[12] Kjell Magne Fauske. *Neural network | TikZ example*. 2006. URL: http://www.texample.net/tikz/examples/neural-network/ (visited on 10/22/2019).

[13] Sorelle A. Friedler et al. "Assessing the Local Interpretability of Machine Learning Models". In: *CoRR* abs/1902.03501 (2019). arXiv: 1902.03501.

[14] Adam Frisk. *What is Project Maven? The Pentagon AI project Google employees want out of*. Apr. 2015. URL: https://globalnews.ca/news/4125382/google-pentagon-ai-project-maven/ (visited on 10/01/2019).

[15] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[16] Felix Grün et al. *A Taxonomy and Library for Visualizing Learned Features in Convolutional Neural Networks*. 2016. arXiv: 1606.07757v1 [cs.CV].

[17] Linwei Hu et al. *Locally Interpretable Models and Effects based on Supervised Partitioning (LIME-SUP)*. 2018. arXiv: 1806.00663 [stat.ML].

[18] Lars Hulstaert. *Interpreting machine learning models*. Feb. 2018. URL: https://towardsdatascience.com/interpretability-in-machine-learning-70c30694a05f.

[19] Lars Hulstaert. *Understanding model predictions with LIME*. July 2018. URL: https://towardsdatascience.com/understanding-model-predictions-with-lime-a582fdff3a3b.

[20] Jacob Kauffmann et al. "From Clustering to Cluster Explanations via Neural Networks". In: (2019). arXiv: 1906.07633. URL: http://arxiv.org/abs/1906.07633.

[21] Min Lin, Qiang Chen, and Shuicheng Yan. "Network In Network". In: *International Conference on Learning Representations*. 2014.

[22] Bernard Marr. *The Wonderful Ways Artificial Intelligence Is Transforming Genomics and Gene Editing*. Nov. 2018. URL: https://www.forbes.com/sites/bernardmarr/2018/11/16/the-amazing-ways-artificial-intelligence-is-transforming-genomics-and-gene-editing/#62fc972242c1/ (visited on 04/10/2019).

[23] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Muller. “Methods for interpreting and understanding deep neural networks”. In: *Digital Signal Processing*. 2018, pp. 1–15.

[24] Grégoire Montavon et al. “Deep taylor decomposition of neural networks”. In: *Proceedings of the ICML 2016 Workshop on Visualization for Deep Learning*. 2016.

[25] Grégoire Montavon et al. “Explaining nonlinear classification decisions with deep Taylor decomposition”. In: *Pattern Recognition*. 2017, pp. 211–222.

[26] Margaret Niess, Shannon Driskell, and Karen F. Hollebrands. *Handbook of research on transforming mathematics teacher education in the digital age*. IGI Global, 2016.

[27] Ruth Omoh. *How Google Is Using AI To Make Voice Recognition Work For People With Disabilities*. June 2019. URL: https://www.forbes.com/sites/ruthumoh/2019/06/10/how-google-is-using-ai-to-make-voice-recognition-work-for-the-disabled/#72804b903c3e (visited on 10/01/2019).

[28] Brett Poulin et al. “Visual explanation of evidence with additive classifiers”. In: *Proceedings of the National Conference on Artificial Intelligence*. Vol. 21. 2. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999. 2006, p. 1822.

[29] *REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL*. Apr. 2016. URL: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex:32016R0679 (visited on 10/01/2019).

[30] Marco Tulio Ribeiro, Sameer, and Carlos Guestrin. *"Why Should I Trust You?": Explaining the Predictions of Any Classifier*. 2016. arXiv: 1602.04938 [cs.LG].

[31] Laura Rieger. “Separable explanations of neural network decisions”. In: *31st Conference on Neural Information Processing Systems*. 2017.

[32] Wojciech Samek, Thomas Wiegand, and Klaus-Robert Muller. “Explainable Artificial Intelligence: Understanding, visualizing and interpreting deep learning models”. In: 2017. arXiv: 1708.08296.

[33] Wojciech Samek et al. “Evaluating the visualization of what a deep neural network has learned”. In: *IEEE transactions on neural networks and learning systems* 28.11 (2016), pp. 2660–2673.

[34] Ramprasaath R Selvaraju et al. “Grad-cam: Visual explanations from deep networks via gradient-based localization”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 618–626. URL: https://ieeexplore.ieee.org/document/8237336.

[35] Ramprasaath Selvaraju R et al. “Grad-CAM: Why did you say that?” In: 2016. arXiv: 1611.07450.

[36] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. “Learning important features through propagating activation differences”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 3145–3153.

[37] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. *Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps*. 2014. arXiv: 1312.6034v2 [cs.CV].

[38] Christian Szegedy et al. “Going Deeper with Convolutions”. In: *CoRR* abs/1409.4842 (2014). arXiv: 1409.4842.

[39] University of Washington (Ribeiro and Guestrin 2016). *LIME*. Version 0.1.1.36. Oct. 2, 2019. URL: https://github.com/marcotcr/lime.

[40] Bill Wilson. *The Machine Learning Dictionary*. 2012. URL: http://www.cse.unsw.edu.au/~billw/mldict.html (visited on 10/11/2019).