1

Explaining the Decisions of Convolutional and Recurrent Neural Networks

Wojciech Samek, Leila Arras, Ahmed Osman, Grégoire Montavon, Klaus-Robert Müller

Abstract

The ability to explain and understand the prediction behaviour of complex machine learning (ML) models such as deep neural networks is of large interest to developers, users and researchers. It allows them to verify the system's decision making and gain new insights into the data and the model, including the detection of its malfunctioning. Moreover, it can also help to improve the overall training process, e.g., by removing detected biases. However, due to the large complexity and highly nested structure of deep neural networks, it is non-trivial to obtain these interpretations for most of today's models. This chapter describes Layer-wise Relevance Propagation (LRP), a propagation-based explanation technique that can explain the decisions of a variety of ML models, including state-of-the-art convolutional and recurrent neural networks. As the name suggests, LRP implements a propagation mechanism that redistributes the prediction outcome from the output to the input, layer by layer through the network. Mathematically, the LRP algorithm can be embedded into the framework of Deep Taylor Decomposition and the propagation process can be interpreted as a succession of first-order Taylor expansions performed locally at each neuron. The result of the LRP computation is a *heatmap* visualizing how much each input variable (e.g., pixel) has contributed to the prediction. This chapter will discuss the algorithmic and theoretical underpinnings of LRP, apply the method to a complex model trained for the task of Visual Question Answering (VQA), and demonstrate that it produces meaningful explanations, revealing interesting details about the model's reasoning. We conclude the chapter by commenting on the general limitations of current explanation techniques and interesting future directions.

1.1 Introduction

Over the years machine learning (ML) models have steadily grown in complexity, gaining predictivity often at the expense of interpretability. Deep neural networks are a prime example of this development. These models typically contain millions of parameters and tens or even hundreds of non-linearly interwoven layers of increasing abstraction. After being fed with vast amounts of data, these models can achieve record performances on complex tasks such as vision (Cireşan et al. 2011; Lu and Tang 2015), language understanding (Bahdanau et al. 2014; Devlin et al. 2018), strategic game playing (Mnih et al. 2015; Silver et al. 2017; Moravčík et al. 2017), medical diagnosis (Esteva et al. 2017; Hannun et al. 2019; Jurmeister et al. 2019) or scientific data analysis (Baldi et al. 2014; Mayr et al. 2016; Schütt et al. 2017). The complexity of state-of-the-art convolutional neural networks (CNNs) can be illustrated on the popular VGG-16 model (Simonyan and Zisserman 2014). This model consists of 16 layers of increasing abstraction and 138M weight parameters, moreover, it requires 15.5G elementary operations (MACs) to classify a single 224×224 image. Clearly, for such a large model it becomes very difficult to *explain* why and how it arrived at its decision, i.e., to find the relevant parts in the input (e.g., pixels), which have triggered an individual decision. Such analysis becomes even more complicated for models with an internal state, e.g., Long Short-Term Memory (LSTM) networks (Hochreiter and Schmidhuber 1997) which are often used to process time series data or textual input, because the internal state influences the model's decision making in complex ways. Unsurprisingly until recently, state-of-the-art CNNs, LSTMs and deep models in general have been commonly regarded as "black boxes".

Although the wish to understand and interpret the decision making process of an AI system is as old as the technology itself, the field of explainable AI (XAI) has seen a significant revival in the last years. With the many advancements in the area of deep learning, also various methods have been recently proposed to make these models more interpretable (Samek et al. 2019). While some of these works aim to construct and train ML models which are by design more interpretable, e.g., by incorporating sparsity priors or disentangling the learned representation, other authors propose methods that are capable of explaining a given (trained) ML model *post-hoc*. This chapter will focus exclusively on the latter methods. We will consider the problem of Visual Question Answering (VQA), where a deep model is trained to answer questions about a given image. In order to do so, the model consists of a LSTM and a CNN part, which process the question and the image, respectively. Thus, explaining decisions of the VQA model requires a XAI technique which can explain both types of models. In this chapter we will introduce such a technique, termed Layer-wise Relevance Propagation (LRP) (Bach et al. 2015). LRP is a generic method to explain individual predictions of ML models by meaningfully redistributing the decisions backwards onto the input variables. The result of this explanation process is a *heatmap* visualizing how much each input variable has contributed to the prediction. In the context of VQA, LRP will thus highlight the relevant words in the question and the part in the image which is relevant for answering the question. Analysing and interpreting these explanations allows us to get insights into the functioning of the VQA model, and in particular helps us understand failure modes of it.

In the remainder of this chapter we motivate the need for explainability (Section 1.2), formalize the explanation problem for simple linear models and discuss the difficulties occurring when extending this concept to deep neural networks (Section 1.3). Then, in Section 1.4 we introduce LRP for convolutional neural networks, demonstrate that it can be theoretically embedded into the framework of Deep Taylor Decomposition (Montavon et al. 2017), and present extensions of the method suited for LSTM architectures. In Section 1.5 we demonstrate experimentally that LRP provides meaningful explanations and helps to understand the reasoning of a model, which was trained on a Visual Question Answering task. We conclude the chapter in Section 1.6 with a discussion of the limitations of current explanation techniques and open research questions.

1.2 Why Explainability ?

This section motivates the need for explainability in machine learning and shows that being able to explain and understand the reasoning of a ML model is highly beneficial for at least three different reasons. For a detailed discussion on the risks and challenges of transparency we refer the interested reader to Weller (2019).

1) Practical advantages of explainability
Explanations can be seen as an additional tool for inspecting and debugging ML models. In this role they help us to identify models which are malfunctioning or have learned a strategy which does not match our expectations. In the extreme case, this may lead to an unmasking of "Clever Hans" predictors[1], i.e., models that rely on spurious correlations and artifacts in the data and do not solve the task they were trained for (Lapuschkin et al. 2019). Such models appear to perform well, but would fail if put to a real test. Explanations largely facilitate quickly identifying such invalid prediction strategies and finding biases in the training data (Anders et al. 2020), something which would be very cumbersome or even practically impossible if using standard performance metrics such as cross-validation. Overall, being able to explain predictions is of high value to ML practitioners.

2) Social and legal role of explainability
The acceptance of AI technology in sensitive domains such as medicine or autonomous driving may very much depend on the ability to explain its decisions. Not only users (e.g., patients) may mistrust a decision or feel uncomfortable if confronted with a black box system, also for experts (e.g., medical doctors) comprehending the reasoning of an AI-based assistant system and being able to verify its prediction is an important requirement for accepting new technologies, in particular if they are the ones who are finally responsible. But even for a perfect AI system, explanations are very valuable, simply because they are part of human interaction and enable the users to make informed decisions. Also from a legal perspective, explainability is of utmost importance, because it concerns anti-discrimination and fairness aspects. For instance, explanations are a direct way to check that the model does not base its predictions on certain features (e.g., age or social status) or that it implements consistent prediction strategies for different groups (e.g., men and women). The EU's General Data Protection Regulation (GDPR) explicitly mentions the *right to explanation* for users subjected to decisions of an automated processing system (Goodman and Flaxman 2017).

3) Theoretical insights through explainability
Explanations provide also new insights into the data and the model. For instance, Lapuschkin et al. (2019) analysed the neural network training process on a reinforcement learning task (Atari Breakout game), and found out that the depth of the architecture and the size of the replay memory both have a strong effect on the effectivity of the network to learn to

[1]Clever Hans was a horse that was supposed to be able to perform simple calculations. However, as turned out later that Hans was not doing the math, but selected the correct answers by watching the body language of the human who asked the questions and presented the candidate answers. Similar behaviours have been reported for a competition-winning ML model, which turned out to classify images of a horse by the presence of a copyright tag (Lapuschkin et al. 2016a); in sequel many further Clever Hans incidents were observed across the board of various deep learning and other ML models (Lapuschkin et al. 2019).

focus on the relevant elements (i.e., ball, paddle) of the game. Besides providing insights into the learned strategies of a model, explanations also allow us to better understand the data. For instance, knowing that the network is focusing on particular features (e.g., genes) when predicting the survival times of a patient, may help to discover unknown causal relationships between these two. The ability of machine learning to uncover such hidden patterns in data has been demonstrated in the past, e.g., in the context of the Go game (Silver et al. 2016). Since explanations make learned prediction strategies accessible for human experts, they have the potential to lead to new scientific insights. Therefore, XAI methods have already been used in various scientific disciplines (e.g., Thomas et al. (2019); Horst et al. (2019); Schütt et al. (2017); von Lilienfeld et al. (2020); Reyes et al. (2018)).

1.3 From Explaining Linear Models to General Model Explainability

This section will introduce the problem of explanation and discuss a simple and mathematically well-founded technique for explaining predictions of linear (and mildly nonlinear) ML models. After explaining why the generalization of this simple XAI technique to deep neural networks fails, we will conclude this section with a brief review of state-of-the-art explanation techniques for deep learning models.

1.3.1 Explainability of Linear Models

Let us consider a simple binary classification setting. Assume we are given a trained ML model with parameters θ, i.e.,

$$f_\theta \colon \mathbb{R}^d \to \mathbb{R}, \tag{1}$$

which estimates the class membership of a given input vector $X = [x_1, \ldots, x_d]^\top \in \mathbb{R}^d$ by the sign of the output, i.e., inputs with $f_\theta(X) > 0$ are classified as "class 1", other inputs are classified as "class 2". In the following, we assume that f_θ is a linear classifier trained to solve the binary classification task. Thus, f_θ has the form

$$f_\theta(X) = \sum_{i=1}^{d} w_i x_i + b, \tag{2}$$

with parameters $\theta = \{w_1, \ldots, w_d, b\} \in \mathbb{R}^{d+1}$.

Problem of Explanation: Explaining the prediction of the model for a specific input X means to determine how much the ith input feature x_i (e.g., pixel) has contributed to the classification decision $f_\theta(X)$. Thus, an *explanation* (or heatmap) is a collection of relevance values of same dimensionality as the input, i.e., $[R_1, \ldots, R_d] \in \mathbb{R}^d$. We want explanations to have some **faithfulness** (Swartout and Moore 1993; Samek et al. 2017) w.r.t. the function $f_\theta(X)$: in other words, each *relevance value* $R_i \in \mathbb{R}$ should indicate how much the ith input feature x_i has contributed to the overall classification decision $f_\theta(X)$. Although there may be different ways to measure such a contribution, the explanation must identify and highlight the features actually used by the model, otherwise it is useless. Consequently, for faithful explanations we may assume that a perturbation of the relevant features has a detrimental effect on the model's prediction (Samek et al. 2017). Specific quantifiable aspects of the explanation that relate to faithfulness are:

1. **Conservation**: The conservation property, e.g. Bach et al. (2015), establishes a connection between the explanation and the model's output. Thus, a conservative explanation allows to interpret the summed up relevance values as the total evidence at the output of the network:

$$\sum_{i=1}^{d} R_i \approx f_\theta(X). \tag{3}$$

   Note that the approximate rather than strict equality accounts for potentially unexplainable elements of the function such as biases in the linear model in Eq. (2).

2. **Model-Data-Interaction**: Furthermore, an explanation should reflect the interaction between the feature and its usage by the model. That means that for one data point $X^{(1)}$ the ith feature may be present and used by the model (i.e., be relevant for the decision), whereas for another data point $X^{(2)}$ it may be present but not used by the model (i.e., be irrelevant for the decision). Thus, in contrast to feature selection (Guyon and Elisseeff 2003) an input feature can not per se be regarded as relevant or irrelevant. This property leads to *individual* explanations.

3. **Signed Relevance**: Finally, we want relevance values to have a meaningful sign. More precisely, a positive R_i should indicate a feature x_i which is relevant to the prediction whereas a negative R_i should mark a feature contradicting the decision. For example, when classify images into urban vs. rest, visual features such as buildings, cars, and pedestrians would be assigned positive relevance scores, whereas trees or wild animals would be assigned negative scores as they tend to contradict the predicted class.

For linear models we can easily find an explanation, which fulfills all the above properties, when defining the contribution of the ith feature as the ith summand in (2), i.e.,

$$R_i = w_i x_i. \tag{4}$$

There are numerous other desirable properties of explanations proposed in the literature (Swartout and Moore 1993; Baehrens et al. 2010; Shrikumar et al. 2016; Montavon et al. 2018; Robnik-Šikonja and Bohanec 2018), however, a commonly accepted mathematical definition of what explanations are and which axioms they need to fulfill is still lacking. Therefore, we do not aim to introduce a rigorous axiomatic definition of an explanation in this chapter. However, we would like to note that many popular explanation techniques do not fulfill the properties described above, even for linear models. For instance, sensitivity analysis (e.g., Baehrens et al. (2010); Simonyan et al. (2013)) computes values R_i indicating how much changes in an input feature translate in changes in the output. For linear models the sensitivity score for the ith input feature is simply w_i, i.e., it is constant with respect to the input X. Thus, sensitivity-based explanations neither reflect the interaction between the feature and the model nor are they convervative or provide a meaningful interpretation of the sign.

1.3.2 Generalizing Explainability to Non-Linear Models

Practical machine learning models are often nonlinear. For example, kernel machines expose the input features through a nonlinear feature map and deep neural networks interleave linear projections with nonlinear activation functions. The resulting output is a complex, interwoven mixture of the input features which are often more capable of learning the prediction task.

When the function is nonlinear but remains locally linear, explanations can be obtained using the well-known Taylor expansion. The latter offers a generic tool to decompose the prediction in terms of elements of a linear sum. If the function $f_\theta : \mathbb{R}^d \to \mathbb{R}$ is twice continuously differentiable at the reference point $\widetilde{X} \in \mathbb{R}^d$, then it can be decomposed as

$$f_\theta(X) = f_\theta(\widetilde{X}) + \sum_{i=1}^{d} \underbrace{(x_i - \widetilde{x}_i) \cdot [\nabla f_\theta(\widetilde{X})]_i}_{R_i} + \varepsilon \tag{5}$$

$$\text{with } \varepsilon = \mathcal{O}(\|X - \widetilde{X}\|^2) \text{ as } X \to \widetilde{X}, \tag{6}$$

where contributions R_i can be identified from the first-order terms, and where ε denotes the higher-order terms. For a well-chosen reference point with $f_\theta(\widetilde{X}) = 0$ (i.e., root point) and small enough higher-order terms, we can explain the predictions of the non-linear model in terms of relevance values R_i, while retaining approximately the conservation property

$$f_\theta(X) \approx \sum_{i=1}^{d} R_i, \tag{7}$$

which we also had for the linear case (up to the linear function's bias term). Note that the solution in (4) can be regarded as a special case of the Taylor-based decomposition with reference point $\widetilde{X} = \mathbf{0}$.

Thus, Taylor expansion offers a generic mathematical tool to decompose a prediction into dimension-wise contributions. But does this method produce meaningful explanations for any nonlinear function? Unfortunately this is not the case. The Taylor approach only provides meaningful decompositions for simple (e.g., locally linear) nonlinear models. It fails to produce meaningful explanations for functions which are highly nonlinear.

For example, the function $f(x, y) = x \cdot y$ is dominated by second-order terms near the origin (x and y only matter jointly). Also for piecewise linear models, e.g., deep neural networks with ReLU activations, this naive Taylor-based decomposition has been shown to provide low-quality explanations (Montavon et al. 2018; Samek et al. 2020). The first problem arises from the complexity of the neural network function and the high-dimensional space to which it is applied. In these high dimensions many root points $\widetilde{X}$ are potentially reachable, however, only a few of them are truly meaningful (close enough, lying on data manifold etc.). Deep neural networks are known to lack robustness to inputs lying outside the data manifold (i.e., adversarial examples), thus selecting a reference point which lies even slightly outside of this complex manifold can lead to unintended behaviour of the model, resulting in noisy and uninformative explanations. For ReLU-based networks without bias, choosing a reference point $\widetilde{X} = \delta X$ is valid[2] for any positive value of δ, and the non-explainable zero-order term

[2]The resulting reference point $\widetilde{X}$ lies on the same linear region as the point X (Montavon et al. 2018).

vanishes in the limit of $\delta \to 0$, hence leading to an explanation that satisfies conservation. We note however that the reference point is in most cases quite far away from the input sample X, and hence, it does not sufficiently contextualize the prediction, causing spurious negative relevance scores. Furthermore, due to the multiscale and distributed nature of neural network representations, its predictions are a combination of local and global effects. The combination of the two effects introduces a nonlinearity, which is impossible to capture by one single linear expansion. Finally, the high complexity of today's deep models often results in a shattered gradients effect (Balduzzi et al. 2017). Thus, methods relying on gradients (as the Taylor expansion in (5)) will systematically produce noisy explanations.

In summary, one can say that the Taylor-based decomposition provides a principled way to decompose a function into dimension-wise contributions, but it only works well for relatively simple (nonlinear) functions.

1.3.3 Short Survey on Explanation Methods

To address the challenge of explaining nonlinear models and overcome the difficulties mentioned above, a number of approaches have been proposed. This section gives a brief overview of the different approaches proposed in the context of deep neural networks. As done in Samek and Müller (2019) we categorize the methods into three classes: surrogate-based, perturbation-based, and propagation-based explanation methods.

We have seen in Section 1.3.1 that simple linear classifiers are intrinsically explainable, because they can readily be decomposed as a sum over individual dimensions. Surrogate-based XAI methods utilize this property and explain predictions of complex classifiers by locally approximating them with a simple surrogate function, which is interpretable. Local Interpretable Model-agnostic Explanations (LIME) (Ribeiro et al. 2016) is the most popular explanation technique falling into this category. While LIME has the advantage of being model-agnostic, i.e., it applies to any black-box model without requiring access to its internal structure, one needs to collect a sample of input-output pairs to build the surrogate. This can be computationally expensive and the result may depend on the sample.

The second class of XAI methods constructs explanations from the model's response to local changes, e.g. some coarse perturbations (Zeiler and Fergus 2014; Zintgraf et al. 2017; Lundberg and Lee 2017), or directly the gradient which can be computed cheaply (Baehrens et al. 2010; Simonyan et al. 2013; Shrikumar et al. 2016). To address the gradient noise and its locality, averaging approaches such as SmoothGrad (Smilkov et al. 2017) or Integrated Gradients (Sundararajan et al. 2017) have been proposed, which improve explanation quality compared to the sole gradient, although averaging incurs an additional computational cost. Additionally, the perturbation approach can also be expressed as an optimization problem (Fong and Vedaldi 2017; Chang et al. 2018; Macdonald et al. 2019), for example, by trying to identify the minimal set of relevant input features, e.g., features which leave the expected classifier score nearly constant when randomising the remaining (non-relevant) features.

Finally, propagation-based methods, such as Layer-wise Relevance Propagation (Bach et al. 2015) explain decisions of neural networks by utilizing the internal structure of the model, specifically, by running a purposely designed backward pass in the network. They give robust explanations and can be computed quickly in the order of a single backward pass. Furthermore, they overcome many of the disadvantages of the other explanation techniques, e.g., they do not require sampling and do not have the problem of gradient shattering. The price to pay for the advantageous properties is the reliance on the structure of the model (i.e., these methods are not model-agnostic and require to carefully design the backward pass to take into account the specificity of each layer in the network). More recent work has focused on systematizing the design of the backward pass (Montavon et al. 2017, 2019) and on extending the approach beyond neural networks (Kauffmann et al. 2019, 2020).

Besides developing novel explanation methods, several works have also focused on comparing existing XAI techniques using objective evaluation criteria. For instance, Samek et al. (2017) measured the quality of explanations using "pixel-flipping". Some authors consider proxy tasks for evaluation (Doshi-Velez and Kim 2017), e.g. using the explanation for localization tasks (Zhang et al. 2016). Other authors evaluated explanations using ground-truth information from a synthetic dataset (Osman et al. 2020). Yet other works assess explanation methods based on the fulfillment of certain axioms (Sundararajan et al. 2017; Lundberg and Lee 2017; Montavon 2019). Although the evaluation of XAI is still an ongoing research topic, propagation-based explanation techniques have already demonstrated their strength in various practical settings.

A recent trend is also to move from individual explanations to *dataset-wide* explanations, that are presented to the user in a way that provides insight into the internal representation or the set of learned behaviours of the classifier. For instance, Lapuschkin et al. (2019) propose a technique for (semi-)automatically searching for interesting patterns in a set of explanations. Other works project explanations of the neural network onto more abstract semantic concepts that can be understood by a human (Bau et al. 2017; Kim et al. 2018) or strive for an interaction with the human user (Baehrens et al. 2010; Hansen et al. 2011), etc.

1.4 Layer-wise Relevance Propagation: Better Explanations by Leveraging Structure

Layer-wise relevance propagation (LRP) (Bach et al. 2015; Montavon et al. 2019) is a XAI method which assumes that the ML model has a layered neural network structure, and leverages this structure to produce robust explanations at low computational cost. LRP implements a reverse propagation procedure from the output of the network to the input features (Lapuschkin et al. 2016b; Alber et al. 2019). The propagation is implemented as a collection of redistribution rules applied at each layer of the network. In contrast to many other XAI methods, which rely on sampling or optimization, LRP computes explanations in the order of one backward pass. This allows for fast GPU-based implementations (Alber et al. 2019) where hundreds of explanations can be computed per second. Furthermore, as we will see in Section 1.4.2, the propagation procedure used by LRP can be embedded in the framework of deep Taylor decomposition (Montavon et al. 2017), which views the backward pass as performing a multitude of Taylor expansions at each layer. Each of these Taylor expansions is simple and tractable, and consequently it conceptually overcomes the various difficulties (finding good reference points, gradient shattering) encountered with the global Taylor expansion approach presented in Section 1.3.2.

By viewing LRP from the perspective of Taylor expansions, it is natural that certain desirable properties of an explanation such as *conservation* will be inherited. In fact, LRP propagation rules will be shown to enforce the conservation *locally* at each neuron, and by extension, this also ensures conservation on a coarser level, e.g. between two consecutive layers, and more globally from the neural network output to the neural network input[3]. Thus, LRP ensures that the total evidence for the decision taken by the model is redistributed, i.e., no evidence is added or removed through the propagation process. Mathematically, we can express this layer-wise conservation property of LRP as

$$f(X) = f(\mathbf{x}^{[0]}) \approx \sum_{k=0}^{n_L} R_k^{[L]} = \ldots = \sum_{i=0}^{n_0} R_i^{[0]}, \tag{8}$$

where $[l]$ and n_l denote the superscript index and input dimension of considered layer l, for $l = 0, \ldots, L$. In the following we show how this redistribution process is implemented in convolutional and recurrent neural networks and how it can be viewed as a layer-wise Taylor expansion of the relevance model $R_k^{[i]} = \widehat{r}_k(\mathbf{x}^{[i-1]})$, i.e., a simple function $\widehat{r}_k$ which computes the relevance of entry/channel k at layer i given the feature map $\mathbf{x}^{[i-1]}$ of layer $i-1$, around a meaningful root point.

1.4.1 LRP in Convolutional Neural Networks

Figure 1.1 illustrates the propagation procedure implemented by LRP for a convolutional neural network. First, the neural network classifies the input, e.g., an image of a rooster. In order to do so, it passes the individual pixel values through a set of convolutional layers, before the resulting activation pattern is classified using fully-connected layers. At every layer i, activations are computed as

$$x_k^{[i]} = \sigma\Big(\sum_{j=1}^{n_{i-1}} x_j^{[i-1]} w_{jk}^{[i]} + b_k^{[i]}\Big), \tag{9}$$

where $\sigma(x) = \max(0, x)$ is the ReLU activation function and the sum runs over all lower-layer activations $x_j^{[i-1]}$, plus an extra bias term $b_k^{[i]}$. The activation(s) at the output layer can be interpreted as the total evidence for the presence of a given class. If we aim to explain the network's decision, then the prediction output (pre-softmax) is used to initialize the last layer's relevance value $R_k^{[L]}$. If the network's prediction is wrong or if we want to investigate the model's view on alternative decisions (e.g., identify features $R_i^{[0]} > 0$ speaking for the presence of a specific class), it may be useful to initialize the last layer's relevance value in a different way (e.g., use the ground truth label). In the example illustrated in Figure 1.1 the classification decision is correct ("rooster"), thus we initialize the last layer's relevance value with the pre-softmax value of the neuron associated with the class "rooster".

In the next step, the backward propagation procedure starts, i.e., the relevance values from the upper-layer are redistributed to the lower-layer neurons. LRP redistributes the

[3]Practically, conservation holds only approximately, due to the possible presence of bias terms at each layer. The latter receive some share of the redistributed evidence (we denote the relevance assigned to the biases at each layer through the 0th summation index in (8)). Also certain LRP rules dissipate relevance by design to improve the signal-to-noise ratio of the explanation (e.g., the LRP-ϵ rule, see later).

relevance values proportionally to the contribution of neurons in the forward pass, i.e.,

$$R_j^{[i-1]} = \sum_{k=1}^{n_i} \frac{z_{jk}^{[i]}}{\sum_{l=1}^{n_{i-1}} z_{lk}^{[i]} + b_k^{[i]}} R_k^{[i]}, \tag{10}$$

where $R_k^{[i]}$ denotes the relevance values of the upper-layer (already known) and $R_j^{[i-1]}$ stands for the newly computed relevance values of the lower-layer neurons. In this generic formula $z_{jk}^{[i]}$ represents the extent to which neuron j from layer $i-1$ has contributed to make neuron k at layer i relevant.

The abstract redistribution rule described in (10) is general enough to be applicable to almost any ML model, including convolutional and recurrent neural networks. In practice, the choice of propagation rule must be done carefully for the explanation method to properly handle the different layer types (Montavon et al. 2018, 2019). Table 1.1 summarizes different redistribution rules proposed for CNNs. Since the last two layers of the network depicted in Figure 1.1 are fully-connected layers, we instantiate the general formulation with the LRP-ϵ redistribution rule. Additionally, for the lower convolutional layers we use a different redistribution rule, namely the LRP-$\alpha\beta$ rule with $\alpha = 1$ and $\beta = 0$. A justification of this layer type specificity of LRP will be provided in Section 1.4.2. The result of the redistribution process is a *heatmap* highlighting the pixels which have contributed the most to the model's decision that the image belongs to the class "rooster". We see that the rooster's head and comb are the most relevant features for this decision. In the following we will discuss specific instances of the redistribution rule for different layer types (fc, conv, input layer) of a convolutional neural network. Section 1.4.3 will separately treat the redistribution of relevance through product layers, which are present in LSTM networks.

The Basic Rule (LRP-0) and Epsilon Rule (LRP-ϵ) (Bach et al. 2015) redistribute the relevance in proportion to two factors, namely the activations of lower-layer neurons and the connection strengths (weights). The intuition behind these rules is that neurons which are more activated during prediction encode something about the input, e.g., features present in the image such as the rooster's comb, and should therefore receive a larger share of relevance than neurons which are not activated (note that with ReLU nonlinearities activations are positive or zero). Since the activation patterns will vary for different inputs, these redistribution rules will produce individual explanations for each input. However, since features may be present but be irrelevant for the task, activations are not the only criterion to guide the relevance redistribution process. The connections between neurons (weights) reflect the integration of the encoded low-level features into higher-level concepts (and finally into the prediction) and should be therefore also taken into account when explaining model decisions. Therefore, both rules redistribute relevance in proportion to the activations of lower-layer neurons and the connection strengths (model-data-interaction). The LRP-ϵ rule includes a small non-zero term ϵ in the denominator. This additional term stabilizes the redistribution process by absorbing some relevance when the contributions to the activation of neuron k are weak or contradictory.

Two rules which distinguish between the supporting (positive relevance) and contradicting (negative relevance) explanatory factors are LRP-$\alpha\beta$ (Bach et al. 2015) and LRP-γ (Montavon et al. 2019). These rules are also based on the strength of the activations and the weight values. By setting the α, β and γ parameters appropriately, we can control how much positive

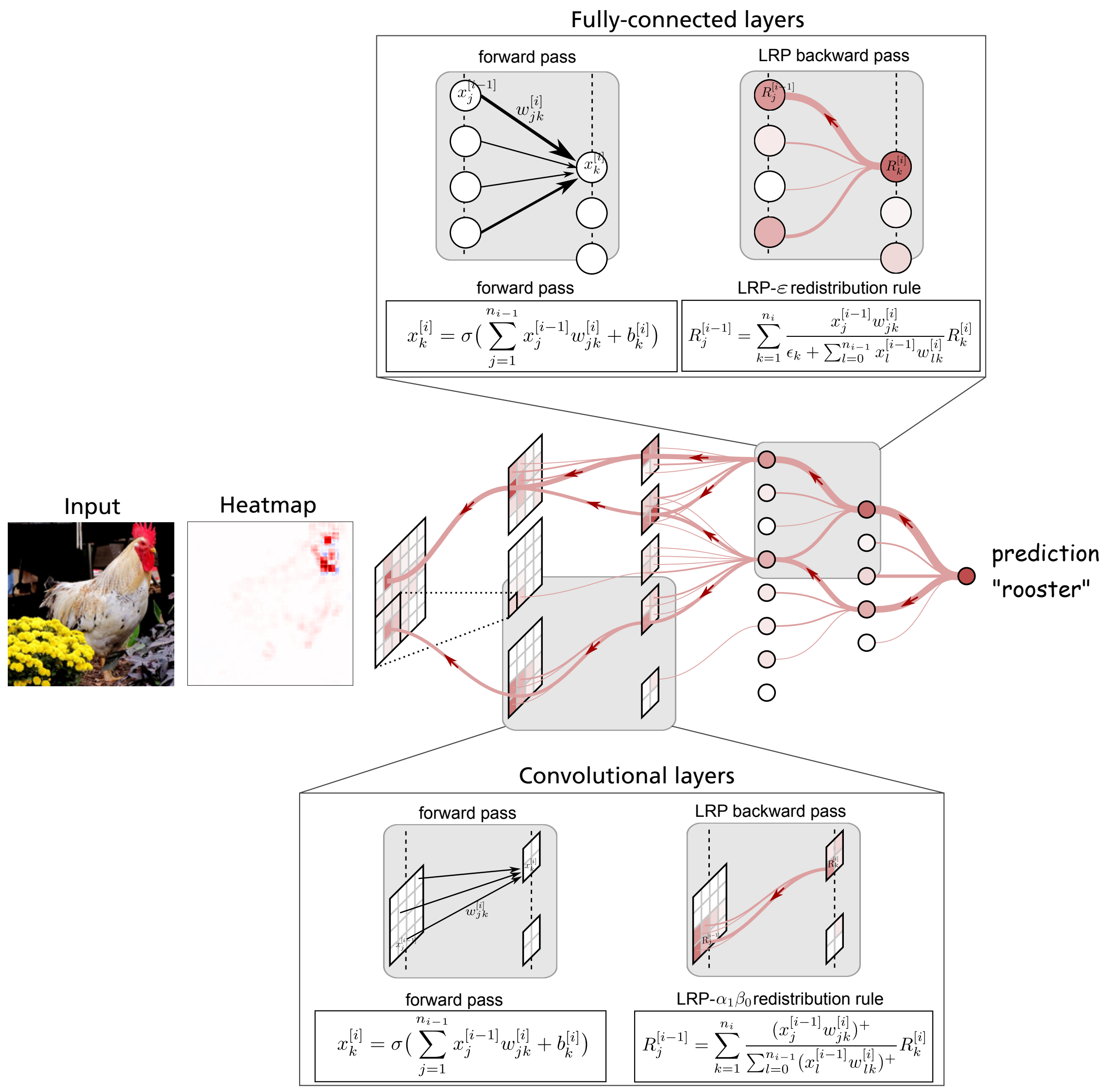

Fully-connected layers
forward pass
LRP backward pass
$x_j^{[i-1]}$
$w_{jk}^{[i]}$
$x_k^{[i]}$
$R_j^{[i-1]}$
$R_k^{[i]}$
forward pass
LRP-ε redistribution rule
$x_k^{[i]} = \sigma(\sum_{j=1}^{n_{i-1}} x_j^{[i-1]} w_{jk}^{[i]} + b_k^{[i]})$
$R_j^{[i-1]} = \sum_{k=1}^{n_i} \frac{x_j^{[i-1]} w_{jk}^{[i]}}{\epsilon_k + \sum_{l=0}^{n_{i-1}} x_l^{[i-1]} w_{lk}^{[i]}} R_k^{[i]}$
Input
Heatmap
prediction
"rooster"
Convolutional layers
forward pass
LRP backward pass
forward pass
LRP-$\alpha_1\beta_0$ redistribution rule
$x_k^{[i]} = \sigma(\sum_{j=1}^{n_{i-1}} x_j^{[i-1]} w_{jk}^{[i]} + b_k^{[i]})$
$R_j^{[i-1]} = \sum_{k=1}^{n_i} \frac{(x_j^{[i-1]} w_{jk}^{[i]})^+}{\sum_{l=0}^{n_{i-1}} (x_l^{[i-1]} w_{lk}^{[i]})^+} R_k^{[i]}$

Figure 1.1: Illustration of the LRP procedure for a CNN. Each neuron redistributes as much relevance to the lower-layer as it has received from the higher layer, i.e., no relevance is lost or artificially added in the explanation process.

relevance is redistributed relative to the negative one (and vice versa). Note that as the γ parameter goes to infinity, LRP-γ approaches LRP-$\alpha\beta$ with $\alpha = 1$ and $\beta = 0$.

The other rules shown in Table 1.1 are flat redistribution and two redistribution rules satisfying different requirements on the first layer. While the flat redistribution rule can be used to reduce the spatial resolution of heatmaps (by simply uniformly redistributing relevance from some intermediate layer onto the input), the $z^{\mathcal{B}}$- and w^2-rule are derived from the deep Taylor decomposition framework (DTD) (Montavon et al. 2017). Almost all redistribution rules in Table 1.1 can be embedded into the DTD framework and interpreted as performing a local Taylor decomposition with specific assumptions on the reference point (see Section 1.4.2). This theoretic foundation of LRP is a strength of the method, because it allows one to design specific optimal rules for different neural network layers (see discussion in Montavon et al. (2019); Kohlbrenner et al. (2020)). Finally, there are two types of layers that are also commonly encountered in practical convolutional neural networks: pooling layers and batch normalization layers. For max-pooling layers, we can redistribute based on a simple winner-take-all scheme. For average pooling layers, we note that they are a particular instance of a linear layer with positive constant weights and therefore, the rules that have been defined for linear layers can also be used. Batch-normalization layers are typically treated by fusing them with the fully-connected or convolution layer they connect to (Montavon et al. 2019). This fusing is done after training but before applying LRP, so that LRP only sees an alternation of convolution/fully-connected, ReLU and pooling layers.

Table 1.1: Overview of different LRP redistribution rules for CNNs. For better readability we include a 0th index term with $x_0^{[i-1]} := 1$ and $w_{0k}^{[i]} := b_k^{[i]}$. We use the following notation $(x)^+ = \max(0,x)$ and $(x)^- = \min(0,x)$.

Name	**Redistribution Rule**	**Usage**	**DTD**
LRP-0 (Bach et al. 2015)	$R_j^{[i-1]} = \sum_{k=1}^{n_i} \frac{x_j^{[i-1]} w_{jk}^{[i]}}{\sum_{l=0}^{n_{i-1}} x_l^{[i-1]} w_{lk}^{[i]}} R_k^{[i]}$	fc layers	✓
LRP-ϵ (Bach et al. 2015)	$R_j^{[i-1]} = \sum_{k=1}^{n_i} \frac{x_j^{[i-1]} w_{jk}^{[i]}}{\epsilon_k + \sum_{l=0}^{n_{i-1}} x_l^{[i-1]} w_{lk}^{[i]}} R_k^{[i]}$	fc layers	✓
LRP-γ (Montavon et al. 2019)	$R_j^{[i-1]} = \sum_{k=1}^{n_i} \frac{x_j^{[i-1]}(w_{jk}^{[i]} + \gamma(w_{jk}^{[i]})^+)}{\sum_{l=0}^{n_{i-1}} x_l^{[i-1]}(w_{lk}^{[i]} + \gamma(w_{lk}^{[i]})^+)} R_k^{[i]}$	conv layers	✓
LRP-$\alpha\beta$ (s.t. $\alpha - \beta = 1$) (Bach et al. 2015)	$R_j^{[i-1]} = \sum_{k=1}^{n_i} \Big(\alpha \frac{(x_j^{[i-1]} w_{jk}^{[i]})^+}{\sum_{l=0}^{n_{i-1}} (x_l^{[i-1]} w_{lk}^{[i]})^+} - \beta \frac{(x_j^{[i-1]} w_{jk}^{[i]})^-}{\sum_{l=0}^{n_{i-1}} (x_l^{[i-1]} w_{lk}^{[i]})^-}\Big) R_k^{[i]}$	conv layers	× (except $\alpha = 1$, $\beta = 0$)
flat (Lapuschkin et al. 2019)	$R_j^{[i-1]} = \sum_{k=1}^{n_i} \frac{1}{n_{i-1}} R_k^{[i]}$	decrease resolution	×
w^2-rule (Montavon et al. 2017)	$R_i^{[0]} = \sum_{j=1}^{n_1} \frac{(w_{ij}^{[1]})^2}{\sum_{l=1}^{n_0} (w_{lj}^{[1]})^2} R_j^{[1]}$	first layer ($\mathbb{R}^d$)	✓
$z^{\mathcal{B}}$-rule (Montavon et al. 2017)	$R_i^{[0]} = \sum_{j=1}^{n_1} \frac{x_i^{[0]} w_{ij}^{[1]} - l_i(w_{ij}^{[1]})^+ - h_i(w_{ij}^{[1]})^-}{\sum_{l=1}^{n_0} x_l^{[0]} w_{lj}^{[1]} - l_l(w_{lj}^{[1]})^+ - h_l(w_{lj}^{[1]})^-} R_j^{[1]}$	first layer (pixels)	✓

1.4.2 Theoretical Interpretation of the LRP Redistribution Process

This section presents the embedding of LRP into the theoretical framework of Deep Taylor Decomposition (DTD) (Montavon et al. 2017) and shows that the redistribution rules introduced in the last section correspond to a particular Taylor expansion performed locally at the neuron. This match between the algorithmic view (LRP) and the theoretic view (DTD) on the explanation problem gives us the possibility to design redistribution rules tailored to each specific layer type of the CNN.

We have seen that Taylor expansion allows to represent a function as a sum of zeroth, first and high order terms (cf. Eq. (5)). If the reference point $\widetilde{X}$ is chosen wisely, one can use the expansion to decompose the function value in terms of dimension-wise contributions (Eq. (7)). However, as discussed in Section 1.3.2 most of such Taylor decompositions are unreliable for deep neural networks due to the gradient shattering problem, the combination of local and global effects and the difficulty to find a good reference point. Thus, although Taylor decomposition is an appropriate mathematical tool to determine dimension-wise contributions, it does not work reliably when trying to explain the deep neural network function in one step, i.e., when trying to linearly approximate its complex input-output relation.

The DTD method, makes use of Taylor expansions, but leverages the deep layered structure of the model. Specifically, DTD uses Taylor expansion not to attribute directly the network output on its input (which we have shown to be instable) but at each layer to attribute the relevance scores $R_k^{[i]}$ to the neurons $\mathbf{x}^{[i-1]}$ in the layer just below. This local attribution task can be interpreted as relevance messages that flow between the neurons of the two consecutive layers.

Technically, DTD provides the theoretical framework for LRP in the following way: (1) the relevance $R_k^{[i]}$ at each layer is mainly driven by local activations and we can therefore approximate it locally using a relevance function $R_k^{[i]} = \widehat{r}_k(\mathbf{x}^{[i-1]})$. (2) First-order Taylor expansion of the relevance function lead to a layer-wise redistribution scheme that corresponds to various LRP rules for appropriate choices of reference points. These two aspects of DTD are described in detail in the following.

Relevance Function

The relevance $R_k^{[i]}$ of neuron k at layer i is a deterministic function r_k of the activations $\mathbf{x}^{[i-1]}$ at lower-level $i-1$, i.e.,

$$R_k^{[i]} = r_k(\mathbf{x}^{[i-1]}). \tag{11}$$

This is due to the fact that both the forward propagation through the neural network as well as the backward relevance redistribution are deterministic processes. We only need to know the activations at layer $i-1$ in order to compute all the activations and relevance values from layer i to the output layer L.

As discussed in Section 1.3.2, Taylor decomposition offers a principled way to determine the dimension-wise contributions of neurons at layer $i-1$ to the relevance value $R_k^{[i]}$. However, the Taylor-based redistribution only works reliably for simple functions $r_k(\mathbf{x}^{[i-1]})$. The $r_k(\mathbf{x}^{[i-1]})$ function is in fact quite complex, because it includes the forward progagation from layer i to the output layer L as well as the relevance redistribution from the output layer L to layer i.

A key insight of DTD is that the relevance function $r_k(\mathbf{x}^{[i-1]})$ can be approximated locally by the simple relevance model $\widehat{r}_k(\mathbf{x}^{[i-1]})$ defined as:

$$\begin{aligned}\widehat{r}_k(\mathbf{x}^{[i-1]}) &= x_k^{[i]} \cdot c_k^{[i]} \\ &= \max\left(0, \textstyle\sum_{j=1}^{n_{i-1}} x_j^{[i-1]} w_{jk}^{[i]} + b_k^{[i]}\right) \cdot c_k^{[i]},\end{aligned} \tag{12}$$

where $c_k^{[i]}$ is a constant. The approximation holds for the last layer L, where the relevance of the output neuron is initialized to be the prediction (pre-softmax) output ($R_k^{[L]} = x_k^{[L]}$). Here the constant $c_k^{[L]}$ is simply 1. Thus, for the last layer we can safely use the Taylor-based redistribution approach.

The question arises whether the approximation still holds for the lower layers, assuming that we have applied LRP in the higher layers. Take for example the LRP-γ rule. If the approximation holds in the layer above, application of LRP-γ result in the relevance scores:

$$\begin{aligned} R_j^{[i-1]} &= \sum_{k=1}^{n_i} \frac{x_j^{[i-1]}(w_{jk}^{[i]} + \gamma(w_{jk}^{[i]})^+)}{\sum_{l=0}^{n_{i-1}} x_l^{[i-1]}(w_{lk}^{[i]} + \gamma(w_{lk}^{[i]})^+)} R_k^{[i]} \\ &= x_j^{[i-1]} \sum_{k=1}^{n_i} (w_{jk}^{[i]} + \gamma(w_{jk}^{[i]})^+) \frac{\max\left(0, \sum_{l=0}^{n_{i-1}} x_l^{[i-1]} w_{lk}^{[i]}\right)}{\sum_{l=0}^{n_{i-1}} x_l^{[i-1]}(w_{lk}^{[i]} + \gamma(w_{lk}^{[i]})^+)} c_k^{[i]} \end{aligned}$$

where we observe that the activation $x_j^{[i-1]}$ is multiplied by a term whose dependence on activations is diluted by two nested sums. This argument supports the modeling of that term as constant in that lower layer as well. Then, by induction, the approximation continues to hold at each layer. A similar argument can be made for the propagation rules LRP-0 and LRP-ϵ.

Taylor-based Redistribution and LRP Rules

Having shown that application of LRP at each layer produces the desired structure of relevance, we now focus on the attribution of this relevance on the layer below by means of a Taylor expansion. The relevance function $\widehat{r}_k(\mathbf{x}^{[i-1]})$ shown in Eq. (12) is a rescaled ReLU neuron taking the lower-layer activations as input. Hence, it consists of two linear pieces corresponding to the activated and deactivated domains. For the deactivated domain, there is no relevance to redistribute. For the activated domain, we consider some reference point $\widetilde{\mathbf{x}}^{[i-1]}$ (also on the activated domain), and a Taylor expansion at this point gives:

$$R_k^{[i]} \approx \widehat{r}_k(\mathbf{x}^{[i-1]}) = \widehat{r}_k(\widetilde{\mathbf{x}}^{[i-1]}) + \sum_{j=1}^{n_{i-1}} \underbrace{(x_j^{[i-1]} - \widetilde{x}_j^{[i-1]}) \cdot [\nabla \widehat{r}_k(\widetilde{\mathbf{x}}^{[i-1]})]_j}_{R_{k\to j}^{[i]\to[i-1]}} + \underbrace{\varepsilon}_{0}. \tag{13}$$

The first-order terms $R_{k\to j}^{[i]\to[i-1]}$ determine how much of $R_k^{[i]}$ should be redistributed on the neuron j of the lower-layer. Thus, this formulation is equivalent to the one in Eq. (5), however, here we determine the relevance redistributed between two adjacent layers (i and $i-1$), whereas in (5) we tried to determine the relevance redistributed between the output and input of the network. The redistribution process between two adjacent layers turns out to be much simpler than the one between output and input (i.e., no gradient shattering, easy to find root point etc.). Note that due to the linearity of the ReLU function on its activated domain (see (12)) higher order terms vanish (i.e., $\varepsilon = 0$). It can be easily seen that the first-order terms reduce to

$$R_{k\to j}^{[i]\to[i-1]} = (x_j^{[i-1]} - \widetilde{x}_j^{[i-1]}) \cdot w_{jk}^{[i]} \, c_k^{[i]}.$$

From this equation, various LRP propagation rules can be recovered. For example, choosing the reference point $\widetilde{\mathbf{x}}^{[i-1]} = \delta \cdot \mathbf{x}^{[i-1]}$ with δ a small positive number gives the LRP-ϵ rule with $\delta = \epsilon(x_k^{[i]} + \epsilon)^{-1}$ (Montavon et al. 2019), we recover the LRP-0 rule in the limit of $\delta \to 0$. The LRP-γ rule is instead obtained by searching the root point on the line $\{\mathbf{x}^{[i-1]} - t\mathbf{x}^{[i-1]} \odot (\mathbf{1} + \gamma \mathbf{1}_{\boldsymbol{w}_k^{[i]} \succeq \mathbf{0}})\}$ (Montavon et al. 2019).

This connection between LRP propagation rules and choices of root points in the DTD framework gives a different perspective on how to choose propagation rules at each layer. These LRP rules no longer appear to be heuristically defined. For example, we can show that LRP-0/ϵ/γ all correspond to a root point whose components are positive. Hence, these rules can be justified as being suitable for use in layers that receive positive quantities, e.g. ReLU activations, as input. Furthermore, LRP rules can be designed directly from the DTD framework, e.g. by choosing root points that satisfy membership to particular domains. For example, the $z^{\mathcal{B}}$-rule and the w^2-rule (shown in Table 1.1) have been originally derived from the DTD framework, where the root point is chosen in the first case to satisfy pixel values box-constraints, or in the second case, not subject to any domain constraint. More details on DTD including proofs can be found in (Montavon et al. 2017).

Choosing LRP Rules in Practice

LRP-0 (Bach et al. 2015) is the simplest rule provided by LRP. It is conservative (except for biases) and gives equal treatment to positive and negative contributions. In practice, LRP-0 leads to explanations that closely follow the function and its gradient (see also Shrikumar et al. (2016); Montavon (2019)). When the network becomes complex and deep, LRP-0 becomes exposed to the problem of shattered gradients, which causes the explanation to become noisy. In the DTD framework, such noisy behavior can be explained by LRP-0 being associated to a root point at the origin, which is far from actual data point, and thus likely to bring irrelevant factors into the explanation. Hence, LRP-0 should be reserved only for simple functions (e.g. the very top layers of a CNN).

The LRP-ϵ rule (Bach et al. 2015) stabilizes the redistribution process by adding a constant to the denominator. This has a sparsification effect, where the relevance of neurons with weak net contributions is driven to zero by the stabilization term. We have observed that the LRP-ϵ works well for redistribution in fully-connected layers present in the top layers of the neural network and also in the top-most convolution layers. In the DTD framework, the gain in stability can be explained by the fact that the root point lies now closer to the data point and thus provides a better contextualization for the explanation.

Convolutional layers of a CNN exhibit strong levels of nonlinearity, especially due to the stacking of many of these layers. In practice, it can be difficult to fully identify the individual positive or negative effect of each pixel. Instead, it is more practical to assign relevance to a collection of pixels, modeling their combined relevance. This behavior of the explanation can be induced by imposing a preference for positive contributions over negative ones. Two rules that impose a different treatment between the positive and negative contributions are LRP-$\alpha\beta$ (Bach et al. 2015) and LRP-γ (Montavon et al. 2019), and they are both suitable for these lower-level layers. LRP-γ in particular, has a DTD interpretation and the corresponding root point is at the ReLU hinge and relatively close to the data point. Hence, like for LRP-ϵ, the benefit of LRP-γ can again be understood as better contextualizing the explanation.

LRP-γ and LRP-$\alpha\beta$ assume positive inputs. This is the case for most convolution layers building on ReLU neurons, but not in the first layer where the input can be e.g. pixel scores that are potentially negative. For these special input layers, purposely designed propagation rules such as the w^2-rule or the $z^{\mathcal{B}}$-rule (Montavon et al. 2017) (cf. Table 1.1) are more suitable. Finally, if we simply want to highlight the receptive fields rather than the contributing features within the receptive field, the 'flat' redistribution rule (Lapuschkin et al. 2019) is appropriate.

Besides, while in this section and section 1.4.1 we highlighted the layer-specific application of the LRP rules on a typical CNN for image classification (according to Montavon et al. (2019); Kohlbrenner et al. (2020)), another configuration of LRP rules might be more appropriate for other network architectures or tasks. For example, the LRP-ϵ rule was found to work well on a word-embedding based CNN for text classification (Arras et al. 2017a). Furthermore, while LRP-$\alpha_1\beta_0$ tends to produce explanations with somewhat too low selectivity on a typical CNN for the task of image classification, the same rule was shown to work very well in the context of explanation-based pruning of CNNs (Yeom et al. 2019), as well as on the CNN based Relation Network model (Osman et al. 2020) we will use for visual question answering in our experimental section 1.5.

More generally, when faced with a novel neural network model or task, and if the network is built upon ReLU activations, then a default configuration that can be tried is to apply the LRP-$\alpha\beta$ rule with $\alpha = 1$, $\beta = 0$ in every hidden layer. LRP-$\alpha\beta$ with parameters fixed to $\alpha = 1$ and $\beta = 0$ considers only positively contributing neurons and delivers positive-valued relevances, and compared to LRP-ϵ and LRP-γ has the advantage of having no free hyperparameter. Furthermore, if an objective assessement of explanation quality is available for a given task, then it is possible to try various combinations of LRP rules and hyperparameters and run an actual hyperparameter selection procedure to optimize explanation for that particular task. In that sense, the multiple hyperparameters provided by LRP can prove very useful to address the exact application's needs.

We now turn to the application of LRP to LSTM networks.

1.4.3 Extending LRP to LSTM Networks

Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber 1997) networks are popular deep models for processing sequential inputs such as biomedical time series, genetic sequences or textual data. These models consist of a *memory cell* storing an internal state. Figure 1.2 displays such a LSTM memory cell. In the forward pass this cell processes the input and sends it through the input gate

$$\mathbf{z}_t = g(\mathbf{W_z}\,\mathbf{x}_t + \mathbf{U_z}\,\mathbf{y}_{t-1} + \mathbf{b_z}) \qquad \text{cell input} \qquad (14)$$

$$\mathbf{i}_t = \sigma(\mathbf{W_i}\,\mathbf{x}_t + \mathbf{U_i}\,\mathbf{y}_{t-1} + \mathbf{b_i}) \qquad \text{input gate} \qquad (15)$$

where $\mathbf{x}_t$ is the input vector at time t, and $\mathbf{z}_t$ and $\mathbf{i}_t$ are the corresponding cell input and input gate activations, respectively. The input gate together with the forget gate control and update the cell state

$$\mathbf{f}_t = \sigma(\mathbf{W_f}\,\mathbf{x}_t + \mathbf{U_f}\,\mathbf{y}_{t-1} + \mathbf{b_f}) \qquad \text{forget gate} \qquad (16)$$

$$\mathbf{c}_t = \mathbf{i}_t \odot \mathbf{z}_t + \mathbf{f}_t \odot \mathbf{c}_{t-1} \qquad \text{cell state} \qquad (17)$$

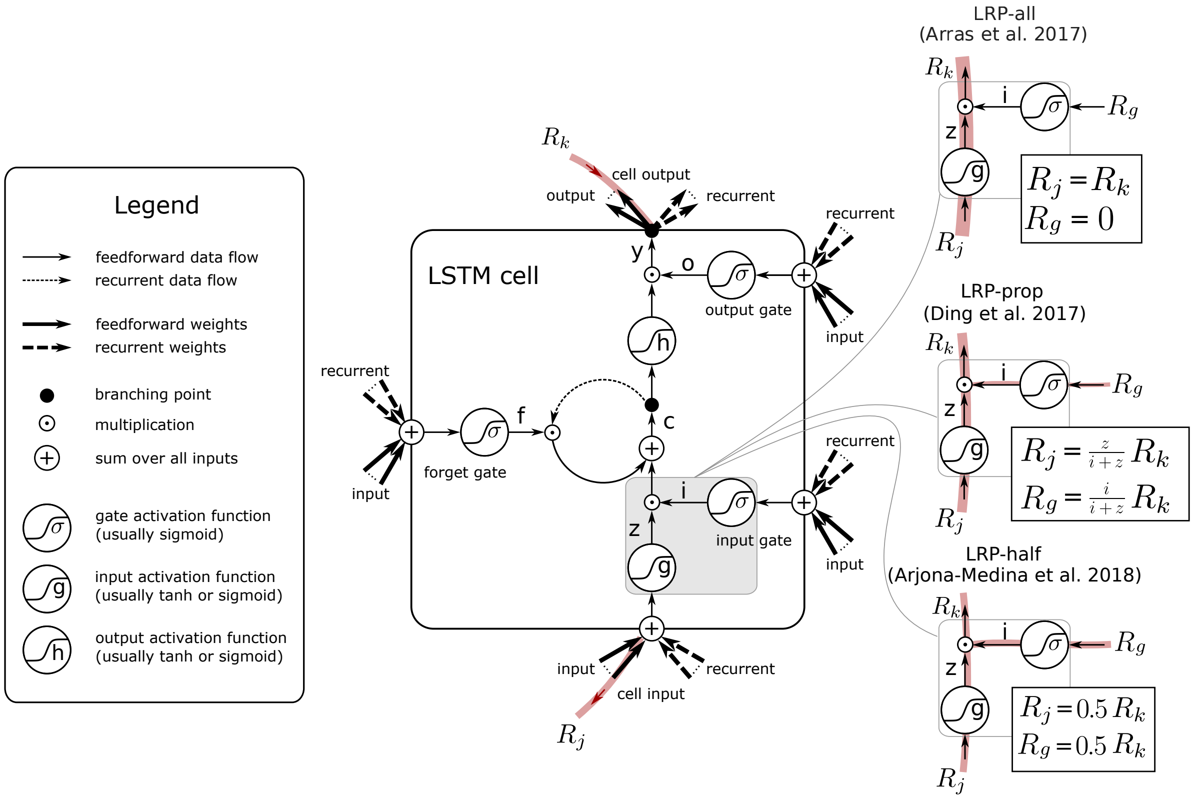

Legend
feedforward data flow
recurrent data flow
feedforward weights
recurrent weights
branching point
multiplication
sum over all inputs
gate activation function (usually sigmoid)
input activation function (usually tanh or sigmoid)
output activation function (usually tanh or sigmoid)
LSTM cell
R_k
output
cell output
recurrent
input
forget gate
output gate
input gate
cell input
R_j
LRP-all (Arras et al. 2017)
$R_j = R_k$
$R_g = 0$
LRP-prop (Ding et al. 2017)
$R_j = \frac{z}{i+z} R_k$
$R_g = \frac{i}{i+z} R_k$
LRP-half (Arjona-Medina et al. 2018)
$R_j = 0.5\, R_k$
$R_g = 0.5\, R_k$

Figure 1.2: Illustration of the LRP procedure for a LSTM. Each neuron, including product neurons, redistributes to the input neurons as much as it has received from the higher layer (sketch of the LSTM cell from (Arras et al. 2019a)).

which itself has an influence on the output of the LSTM cell

$$\mathbf{o}_t = \sigma\left(\mathbf{W_o}\, \mathbf{x}_t + \mathbf{U_o}\, \mathbf{y}_{t-1} + \mathbf{b_o}\right) \qquad \text{output gate} \qquad (18)$$
$$\mathbf{y}_t = \mathbf{o}_t \odot h\left(\mathbf{c}_t\right) \qquad \text{cell output} \qquad (19)$$

Thus, the LSTM output $\mathbf{y}_t$ depends on the actual input $\mathbf{x}_t$ as well as the cell state $\mathbf{c}_t$. Both factors are connected through a point-wise multiplication (denoted as $\odot$). The activation functions g, h are typically tanh or sigmoid, and the gate activation σ is a sigmoid.

For applying the LRP propagation principle to LSTMs, the same redistribution rules which were proposed for convolutional neural networks with ReLU activation (as introduced in Table 1.1) can also be employed in the LSTM to redistribute the relevance through linear layers followed by an element-wise activation (even though the LSTM network typically uses different activation functions than ReLU). In particular, previous works relied on the LRP-ϵ rule for that purpose (Arras et al. 2017b; Ding et al. 2017; Arjona-Medina et al. 2019), i.e., the rule which is recommended for dense layers in CNNs, and which provides a signed relevance. This means that in practice linear layers *redirect* the higher-layer relevance proportionally to the neuron forward pass contributions (i.e., the neuron's activated value times the connection weight), while through element-wise activation layers the relevance is backward propagated identically (no relevance *redirection* through such layers).

Besides, a novel challenge arises when explaining LSTM networks (as well as other gated recurrent networks): it comes from the point-wise multiplicative connections. To handle such non-linearities in the LRP backward redistribution process, authors have proposed several redistribution rules verifying the layer-wise relevance conservation property of LRP (Arras et al. 2017b; Ding et al. 2017; Arjona-Medina et al. 2019). Denoting by x_k such a product neuron in the forward pass, we note that it has the form

$$x_k = x_g \cdot x_j, \tag{20}$$

where x_g denotes a sigmoid activated *gate* neuron, and x_j, the remaining neuron, is a *signal* neuron. Per design of the LSTM network (Hochreiter and Schmidhuber 1997), the role of the gate is to control the flow of the information in the LSTM cell (let the information pass if open, or don't pass if closed), while the signal neuron carries the information itself; this information is either conveyed by the cell input, or stored in the LSTM cell's memory via the cell state. A similar configuration for products can also be found in other popular gated recurrent networks, like e.g., GRUs (Cho et al. 2014).

Given the upper-layer relevance R_k of the neuron x_k, three alternatives rules can be used to redistribute this quantity onto the input neurons x_g and x_j. These rules are illustrated in Figure 1.2 (right), for the particular case of the product between input gate i and cell input z.

LRP-all: One rule proposed by Arras et al. (2017b) assigns all the relevance to the signal neuron, i.e., $R_j = R_k$ and $R_g = 0$. This redistribution follows a signal-take-all strategy. It is based on the idea that gates could be considered as connection weights (although their value is not constant). Accordingly, they influence the LRP redistribution process through the computation of R_k (which, via the relevance redistribution in the next higher linear layer, is proportional to x_k and thus also depends on x_g), but they don't get a relevance value assigned *per-se*, since their expected role in the LSTM forward pass is only to *reweight* the value of the signal x_j.

LRP-prop: The rule proposed by Ding et al. (2017) redistributes the relevance proportionally to the neurons activated values, i.e., $R_j = \frac{x_j}{x_j + x_g} R_k$ and $R_g = \frac{x_g}{x_j + x_g} R_k$.

LRP-half: A rule proposed by Arjona-Medina et al. (2019) redistributes the relevance equally, i.e., $R_j = R_g = 0.5 R_k$.

On commonly used LSTMs (Greff et al. 2017) that follow the structure given by equations (14)-(19) and that use the tanh non-linearity for the cell input and output (functions g and h in (14) and (19)), Arras et al. (2019b,a) carefully compared these three redistribution rules in simulations and experiments with real-world NLP data. The qualitative and quantitative results showed a clear superiority of the LRP-all rule[4] (Arras et al. 2017b) over LRP-prop and LRP-half. Independent works also successfully applied the LRP-all redistribution rule

[4]Note though that on non-standard customized LSTM models, as were introduced in Arjona-Medina et al. (2019); Arras et al. (2019a), other LRP rules for product neurons can become advantageous over LRP-all.

to LSTMs on a synthetic task, in the medical domain (Yang et al. 2018), as well as in NLP (Poerner et al. 2018) and in computer security (Warnecke et al. 2020), demonstrating the resulting LRP explanations are superior to other explanations methods for recurrent neural networks.

DTD Interpretation

In the following we will show that the LRP-all rule can further be motivated under the theoretical framework of Deep Taylor Decomposition (Montavon et al. 2017), as was proposed in Arras et al. (2019a). Let us introduce the neuron pre-activations z_g and z_j, for the neurons x_g and x_j, respectively, and consider a product of neurons where the signal is tanh activated

$$x_k = \mathrm{sigm}(z_g) \cdot \tanh(z_j).$$

Suppose the relevance of the product neuron $R_k(z_g, z_j)$, as a function of pre-activations, can be approximated by a simple relevance model of the form:

$$\widehat{R}_k(z_g, z_j) = \mathrm{sigm}(z_g) \cdot \tanh(z_j) \cdot c_k = x_k \cdot c_k,$$

where c_k is a constant, such that $R_k(z_g, z_j) = \widehat{R}_k(z_g, z_j)$ locally. This is the generic relevance form assumed by the DTD framework (Montavon et al. 2017). By performing a Taylor expansion of this relevance model at a root point $(\widetilde{z}_g, \widetilde{z}_j)$, we obtain:

$$\begin{aligned} \widehat{R}_k(z_g, z_j) &= \widehat{R}_k(\widetilde{z}_g, \widetilde{z}_j) && (= 0) \\ &+ \mathrm{sigm}'(\widetilde{z}_g) \cdot \tanh(\widetilde{z}_j) \cdot c_k \cdot (z_g - \widetilde{z}_g) && (= R_g) \\ &+ \mathrm{sigm}(\widetilde{z}_g) \cdot \tanh'(\widetilde{z}_j) \cdot c_k \cdot (z_j - \widetilde{z}_j) && (= R_j) \\ &+ \epsilon \end{aligned}$$

Using the nearest root point in the space of pre-activations (Arras et al. 2019a), which is given by $\widetilde{z}_j = 0$ and $\widetilde{z}_g = z_g$, it follows the LRP-all rule: $R_g = 0$ and $R_j = \widehat{R}_k$. Moreover, using this root point, higher order terms of the form $(z_g - \widetilde{z}_g)^k$ as well as the interaction terms $(z_g - \widetilde{z}_g)^{\cdots}(z_j - \widetilde{z}_j)^{\cdots}$ vanish, thus the error term ϵ only depends on the signal variable z_j. In other words, if the *tanh* activation is working near its linear regime, then the error term ϵ in the LRP decomposition is negligible (no matter whether the gate's *sigmoid* activation is saturated or not). Lastly, if instead of using a *tanh* activation for the signal neuron, one would use a ReLU activation for example, then the approximation error would be exactly zero in this case (i.e., $\epsilon = 0$). For more information on the various DTD relevance models for different signal activation functions we refer to (Arras et al. 2019a).

1.5 Explaining a Visual Question Answering Model

This section demonstrates the ability of LRP to explain the decisions of a complex model trained for visual question answering (VQA) with the CLEVR dataset (Johnson et al. 2017a). VQA is a multi-modal task at the intersection between nature language processing and computer vision: the model is fed with an image and a textual question about that image, and is asked to predict the answer to that question, either with a single word or in free-form text.

Dataset and Model

The CLEVR dataset is a synthetic VQA task which was proposed by Johnson et al. (2017a) to avoid the biases present in VQA benchmarks based on real-world images (Antol et al. 2015; Goyal et al. 2017): its primary goal was to diagnose the visual reasoning capabilities of state-of-the-art VQA models. It is based on images of 3D rendered objects, more precisely geometric shapes positioned on a plane surface with a grey background. Each object has 4 types of attributes (among 8 colours, 2 materials, 2 sizes, and 3 shapes). The questions are designed to test the following reasoning abilities: spatial relationships between objects, comparison and recognition of object attributes, object counting, comparison of object sets sizes. The data consists of 70.000/15.000/15.000 training/validation/test images, and resp. 699.989/149.991/149.988 questions (i.e. there are roughly 10 questions per image). The prediction problem is framed as a 28-way classification task (the output is a single word among a vocabulary of size 28).

Early work on this task used complex models, e.g., with stacked attention (Yang et al. 2016) or with an explicit representation of the question generation program (Johnson et al. 2017b), but Santoro et al. (2017) proposed a simpler architecture based on a Relation Network (RN) that performs even better in terms of prediction accuracy. We re-implement their model and train it as described in (Santoro et al. 2017). Our trained model reaches a test accuracy of 93,3% on the CLEVR dataset (the original authors report a performance of 95,5%).

The model architecture is displayed in Fig. 1.3. It consists of a 4-layer CNN to extract feature maps from the image, a LSTM network to process the question, and a pairwise concatenation of visual "objects" together with the question representation to fuse the visual and textual information. Here the "objects" are simply pixels in the CNN last layer feature maps. Each layer of the CNN has the following structure: `conv` $\rightarrow$ `relu` $\rightarrow$ `batchnorm`, with 24 kernels of size 3×3 and stride 2 (no padding). The LSTM part of the network is a unidirectional LSTM with word embeddings of size 32, and a hidden layer of size 128. The paired representations (object pair concatenated with the LSTM final hidden state) are passed to a Relation Network (RN), which is made of a 4-layer MLP of fully-connected layers of size 256, each followed by ReLU activation, and a final element-wise summation layer. The resulting representation is passed to a 3-layer MLP with fully-connected layers of size 256, followed each by ReLU activation. The output layer is a fully-connected layer of size 28. For preprocessing the questions, we removed punctuation and performed lowercasing, the resulting input vocabulary has size 80. For preprocessing the images, we resized them to 128 $\times$ 128 pixels, and rescaled the pixel values to the range [0, 1]. Training was done with the Adam optimizer.

Methods for Explaining VQA

In order to get insights into the model's prediction strategies, we compute relevance values for the question (explaining the LSTM part of the model) and the image (explaining the CNN part of the model) using LRP. For the LSTM part of the network, we employ the LRP-all rule for product layers, and the LRP-ϵ rule (with $\epsilon = 0.001$) for remaining layers. For the CNN and RN parts of the network (i.e., on the image processing side), we use the LRP-$\alpha\beta$ rule with $\alpha = 1$ and $\beta = 0$, as this variant was shown to perform best according to

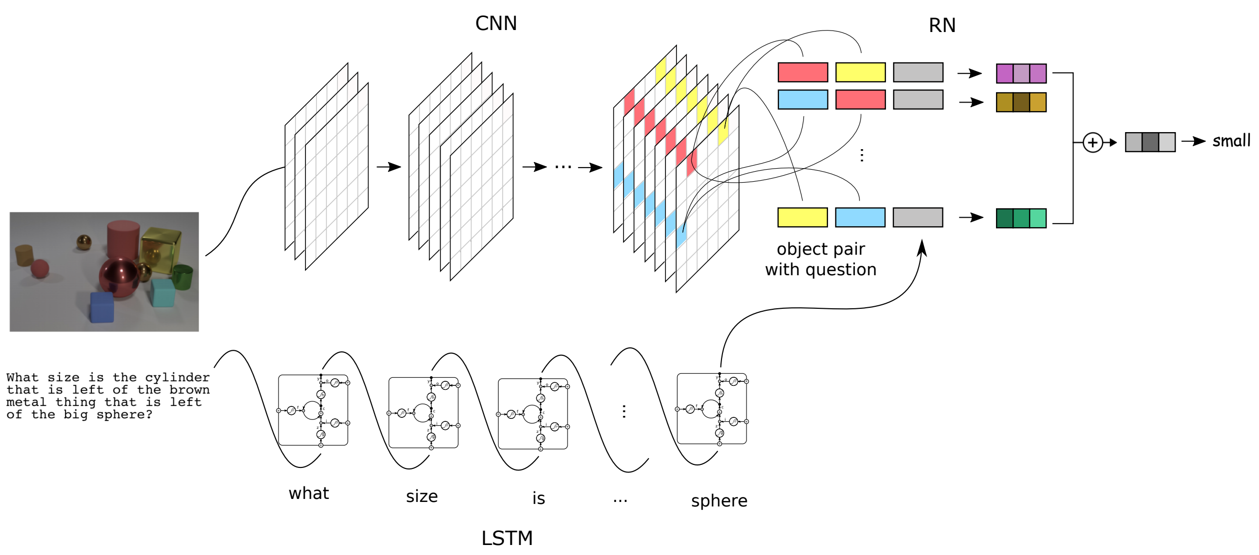

CNN
RN
object pair with question
small
What size is the cylinder that is left of the brown metal thing that is left of the big sphere?
what
size
is
...
sphere
LSTM

Figure 1.3: VQA model from (Santoro et al. 2017) used in our experiments.

quantitative experiments performed in (Osman et al. 2020), in comparison to a composite application of the LRP-$\alpha\beta$/LRP-ϵ rules for different layers.

In addition, we compute heatmaps using two simple baseline methods, namely Gradient $\times$ Input (Shrikumar et al. 2016) and Occlusion (Li et al. 2016). The first method computes the relevance of an input feature by simply using the partial derivative of the prediction function (pre-softmax) multiplied by the feature's value, i.e., $R_i^{[0]} = \frac{\partial f_c}{\partial x_i^{[0]}}(\mathbf{x}^{[0]}) \cdot x_i^{[0]}$ (where f_c is the prediction function for the target class c, and $\mathbf{x}^{[0]}$ is the input), whereas the latter method computes the relevance value using a difference of probabilities $R_i^{[0]} = P_c(\mathbf{x}^{[0]}) - P_c(\mathbf{x}_{|x_i^{[0]}=0})$ (where P_c are the predicted softmax probabilities, respectively, on the original unmodified input, and on the input where the feature of interest $x_i^{[0]}$ has been set to zero).

For all methods we explain the decision for the model's *predicted* class. For the question heatmaps, in order to get one relevance value per word, we sum up the relevances over the word embedding dimensions, and take the absolute value for visualization. For the image heatmaps, we sum up the relevances across the channels, take the absolute value and apply gaussian blurring (with standard deviation 0.02 times the image size) for visualization of the original image overlayed with the heatmap. For raw heatmap visualization we just sum the relevances across the channels.

Results and Insights

In order to see which words are the most relevant per question type, we perform a word relevance statistic over the CLEVR validation set. For each question we select the 3 most relevant words, and compile selected words per question type by frequency. Then we retrieve the words with the highest frequencies for a few question types in Fig. 1.4. We manually highlighted words that are directly related to the question type, regardless of the image, in green. Note that the network does not have access to the question types during prediction.

One can see that the nouns `shape` and `colour` have been identified (except by Gradient

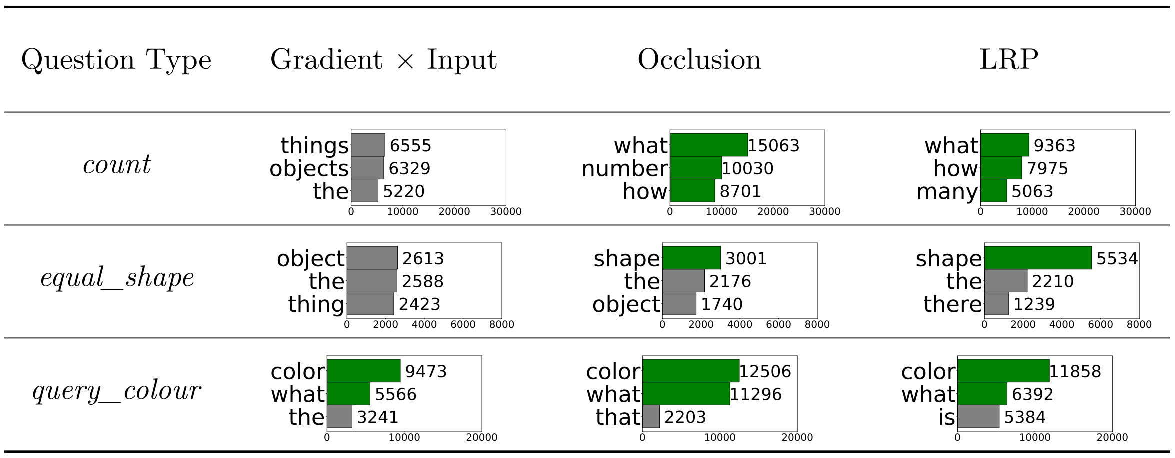


Figure 1.4: Most important words for three CLEVR question types, sorted by decreasing frequency over the validation set. A word is picked as important if it appears in the top 3 relevant words of a question.

× Input) to be relevant to answer questions about the shape or colour of objects. For the question type *count*, where the goal is to count objects in a set, Occlusion and LRP both selected meaningful words, while Gradient × Input attributes high relevance to generic words like `the`, `things`, `objects`. Similar words were selected by Gradient × Input for the question type *equal_shape*, which suggests that the latter method is less suited to find important words related to the question type.

Additionally, we visualize relevance heatmaps for individual data points in Fig. 1.5 and Fig. 1.6, using Gradient × Input and LRP (we did not compute heatmaps for Occlusion, since this method is very expensive to compute on the image side). The exemplary data points were automatically selected in the following way: we conducted a search over both correctly and falsely predicted CLEVR validation points with specific question types, and retrieved the three points with the highest predicted probabilities, i.e., the points where the model is very confident with its prediction. Indeed we expect the corresponding explanations to be more focused and insightful on such data points, while on data points where the model is hesitating the heatmaps might be more diffuse and less informative. For correctly predicted data points (Fig. 1.5), we considered all questions that query an object's attribute, i.e., we used the question types *query_material*, *query_colour*, *query_size*, *query_shape*. For falsely predicted data points (Fig. 1.6) we used only the question type *query_colour*.

From the heatmap visualizations of the textual questions in Fig. 1.5 and Fig. 1.6 we can not make a clear-cut statement about which explanation method is qualitatively better. Both methods seem to deliver equally sparse explanations. When the questions are about an object's material (Fig. 1.5), the word `material` is often highlighted (except in the last question for LRP); when they are about a colour (Fig. 1.6), the word `colour` is always highlighted by LRP, and highlighted once by Gradient × Input. However it seems that Gradient × Input attributes higher relevances to the last words in the question compared to the question's beginning (which is probably due to a vanishing gradient effect induced by the forget gate in the unidirectional LSTM), while LRP is able to assign a high relevance, e.g., to the word `colour` (Fig. 1.6), even when it appears at the question's beginning.

On the image side however, we clearly see that the LRP heatmaps in Fig. 1.5 and Fig. 1.6

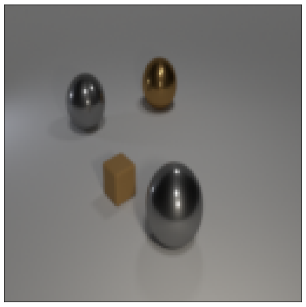
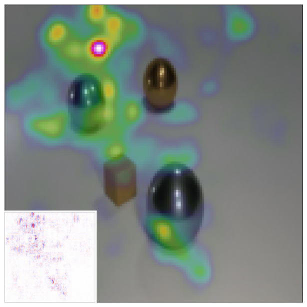
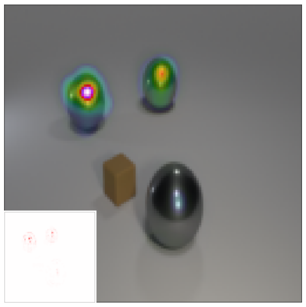
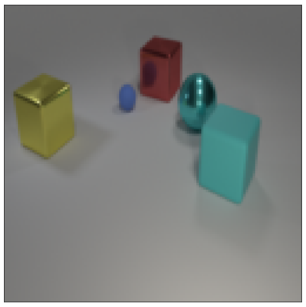
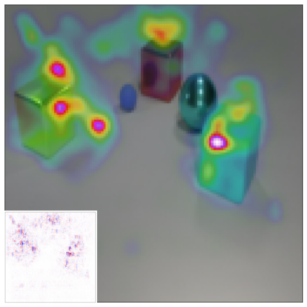
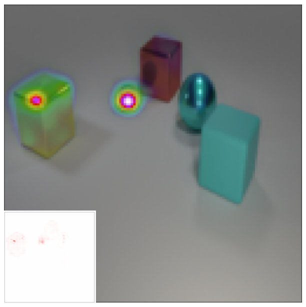
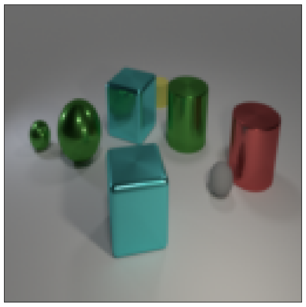
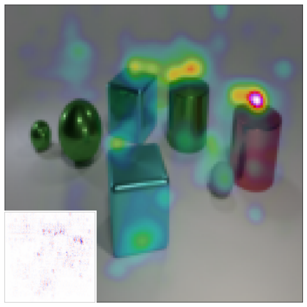
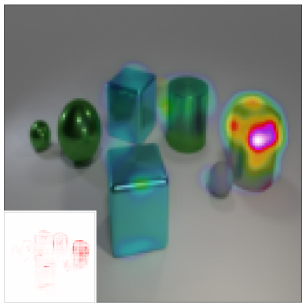
Question	Answer	Gradient × Input	LRP
`there is a large sphere on the left side of the big brown shiny object; what material is it ?`		`there is a large sphere on the left side of the big brown shiny object; what material is it ?`	`there is a large sphere on the left side of the big brown shiny object; what material is it ?`
	**True:** *metal* **Predicted:** *metal*		
`there is a big thing left of the small sphere; what material is it ?`		`there is a big thing left of the small sphere; what material is it ?`	`there is a big thing left of the small sphere; what material is it ?`
	**True:** *metal* **Predicted:** *metal*		
`there is a large object right of the rubber thing that is in front of the big cyan metallic cube behind the tiny gray matte sphere; what is its material ?`		`there is a large object right of the rubber thing that is in front of the big cyan metallic cube behind the tiny gray matte sphere; what is its material ?`	`there is a large object right of the rubber thing that is in front of the big cyan metallic cube behind the tiny gray matte sphere; what is its material ?`
	**True:** *metal* **Predicted:** *metal*		

Figure 1.5: Heatmaps for three CLEVR questions which are correctly predicted. On the image-side, we visualize the heatmap overlayed with the original image, as well as the raw heatmap (on bottom left).

are qualitatively better than those delivered by Gradient × Input: the LRP explanations

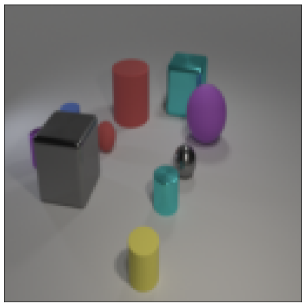
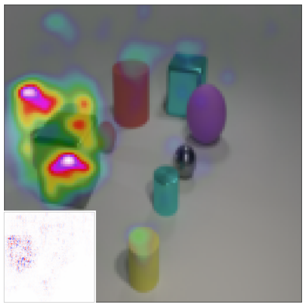
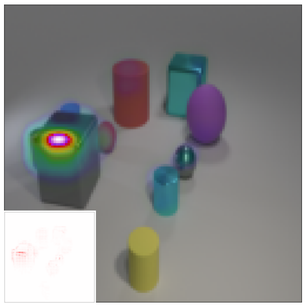
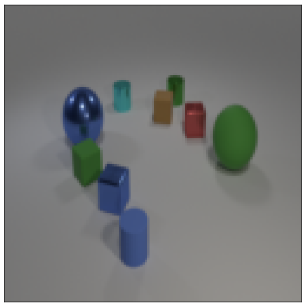
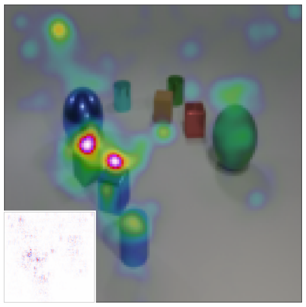
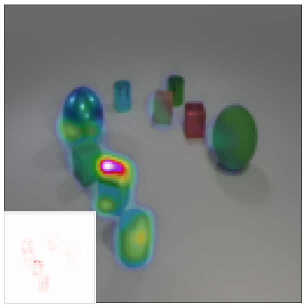
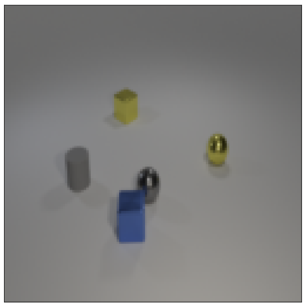
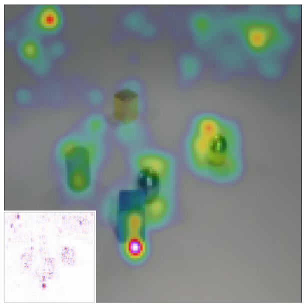
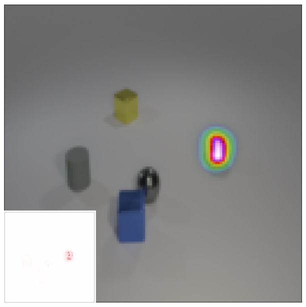
Question	Answer	Gradient × Input	LRP
the metal object on the left side of the blue cylinder is what color ?		the metal object on the left side of the blue cylinder is what color ?	the metal object on the left side of the blue cylinder is what color ?
	**True:** *purple* **Predicted:** *gray*		
what is the color of the object that is both on the left side of the cyan metal cylinder and in front of the large green rubber ball ?		what is the color of the object that is both on the left side of the cyan metal cylinder and in front of the large green rubber ball ?	what is the color of the object that is both on the left side of the cyan metal cylinder and in front of the large green rubber ball ?
	**True:** *green* **Predicted:** *blue*		
what is the color of the object that is to the right of the blue object and in front of the tiny gray matte cylinder ?		what is the color of the object that is to the right of the blue object and in front of the tiny gray matte cylinder ?	what is the color of the object that is to the right of the blue object and in front of the tiny gray matte cylinder ?
	**True:** *gray* **Predicted:** *yellow*		

Figure 1.6: Heatmaps for three CLEVR questions which are falsely predicted. On the image-side, we visualize the heatmap overlayed with the original image, as well as the raw heatmap (on bottom left).

are generally less noisier, more concentrated on objects, and provide pertinent clues to the model's visual reasoning (as we will see below), while Gradient × Input seems to attribute (partly) spurious relevances to background areas.

Let us take a closer look at the LRP heatmaps in Fig. 1.5. In the first question, the target object of the question is the grey metal sphere at the top left of the scene. In the VQA question this object is referenced through its spatial location relatively to the brown sphere. And correspondingly, the LRP visual explanation both highlights the target object, as well as the referring object of the question. The second question has a similar pattern, where the target object is the yellow cube, and the referring object is the small blue sphere. Again the LRP heatmap reveals the two important objects of the question. On the contrary the Gradient $\times$ Input heatmaps are not helpful to understand the model's decisions for question 1 and 2. Finally in the third question, the target object of the VQA question is the red cylinder. Here the spatial relationships formulated in the VQA question are more intricate and involve multiple objects, including a big cyan cube and a small rubber sphere. Thus the LRP explanation is more diffuse: it highlights several objects, namely the two cyan cubes as well as all objects on the right of them. Intuitively it makes sense that the model is focusing on all these objects to answer the given more complex question. Also the target object of the question (the red cylinder) is identified by the LRP heatmap.

Other useful insights can be revealed by the LRP explanations to understand misclassified examples in Fig. 1.6. In these questions the colour of a given object is asked. And in all cases, the object with the highest relevance, as identified by the LRP heatmap, is consistent with the model's predicted answer, while the Gradient $\times$ Input heatmaps are less distinct. In the first question, the target object of the question (the small purple object left of the grey cube), as well as the referring object (the small blue object behind the grey cube) are highly occluded by the grey cube, which explains why the model did not correctly identify the target object of the question and made a false prediction by focusing on the grey cube instead. In the second question the target object of the VQA question is the green cube, however, according to the LRP heatmap, the model identified instead the blue cube as being the target object of the question. Thus the model mistakenly interpreted the spatial relationships defined by the VQA question. A similar phenomenon can be observed in the third question: the target object of the VQA question is the grey metal sphere, but the LRP heatmap reveals the model identified instead the yellow metal sphere as being the target object, which again confirms the model has some difficulty to catch some subtle spatial relationships, here obviously it did not correctly recognise the "in front of" relationship.

Overall, the LRP visual explanations helped to understand *why* the VQA model generated a particular answer. A further interesting point to note is that the considered neural network model we use (which is based on a Relation Network (Santoro et al. 2017)) does not contain any explicit attention mechanism in its architecture. Still the LRP heatmaps could reveal image regions that are important to the model's decision, and that resemble typical attention heatmaps (as provided, e.g., by highly customized models towards the CLEVR dataset like in Mascharka et al. (2018)).

1.6 Discussion

This chapter has presented LRP as a powerful technique to explain predictions of complex ML models such as convolutional and recurrent neural networks. The redistribution rules of LRP have been derived from first principles using the Deep Taylor Decomposition framework.

It has been shown that a careful choice of the redistribution rules (respectively root points in DTD) is very important when applying LRP to composite architectures, i.e., models combining layers with different properties. A naive choice of redistribution rule, e.g., LRP-ϵ for all layers of an image classification CNN, or LRP-prop for a standard LSTM model, can produce poor results. However, when applied properly, LRP can produce meaningful explanations and lead to interesting insights, even for complex and multimodal models such as the one used for the VQA task.

Besides the successful application of LRP to different scenarios, progress has been recently also achieved on the development of criteria to objectively assess the quality and utility of explanations (Samek et al. 2017; Lundberg and Lee 2017; Montavon 2019; Osman et al. 2020). Here several metrics as well as axiomatic approaches have been developed. Still many challenges exist. For instance, while several general frameworks for XAI have been developed, e.g., based on Deep Taylor Decomposition (Montavon et al. 2017), Shapley values (Lundberg and Lee 2017) or rate-distortion theory (Macdonald et al. 2019), a unified theory of explanation is still lacking.

A limitation of today's methods is the low semantic level of explanations. For instance, heatmaps do not allow to distinguish whether a group of features is jointly relevant for a given classification decision or whether each feature in this group contributes individually. Pixel-level heatmaps also do not provide any information about the underlying concepts in the data, e.g., objects in the image. The resulting interpretation gap (i.e., relevant pixels to relevant object) has to be closed by the recipient of explanations, which can be difficult and erroneous. Also the vulnerability of explanation to adversarial manipulations (Dombrowski et al. 2019) is a serious challenge, because it may undermine trust in the explanation results.

Promising future research topics in the field of XAI are the use of explanations beyond interpretability. For instance, a recent paper by Yeom et al. (2019) demonstrates that LRP relevance scores can be used to prune a neural network. Other applications of XAI, e.g., in the detection of adversarial attacks or general model improvement (e.g., by self-supervision) are interesting future research directions. Also the field of human-machine interaction offers various opportunities for explainability research. For instance, questions such as "what type of explanation is most useful for human", "how can we move towards interactive explainability" or "how can we prevent that misleading explanations harm performance" need to be investigated in order to allow for reliable and efficient human-machine interaction, see e.g., Baehrens et al. (2010); Hansen et al. (2011) for early studies in this spirit. Finally, bringing the concepts of XAI to other types of models (Lundberg et al. 2020) or ML tasks (Kauffmann et al. 2019) remains an active area of research.

Acknowledgement. This work was supported by the German Ministry for Education and Research (BMBF) as BIFOLD - Berlin Institute for the Foundations of Learning and Data (ref. 01IS18025A and ref. 01IS18037A). KRM also received support from the Institute of Information & Communications Technology Planning & Evaluation (IITP) grants funded by the Korea Government (No. 2019-0-00079, Artificial Intelligence Graduate School Program, Korea University).

Bibliography

Alber, M., Lapuschkin, S., Seegerer, P., Hägele, M., Schütt, K. T., Montavon, G., Samek, W., Müller, K.-R., Dähne, S., and Kindermans, P.-J. (2019). iNNvestigate neural networks! *Journal of Machine Learning Research*, *20*(93), 1–8.

Anders, C. J., Neumann, D., Marinc, T., Samek, W., Müller, K.-R., and Lapuschkin, S. (2020). XAI for Analyzing and Unlearning Spurious Correlations in ImageNet. *ICML 2020 Workshop - XXAI: Extending Explainable AI Beyond Deep Models and Classifiers*.

Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C. L., and Parikh, D. (2015). VQA: Visual Question Answering. In *International Conference on Computer Vision (ICCV)*, (pp. 2425–2433).

Arjona-Medina, J. A., Gillhofer, M., Widrich, M., Unterthiner, T., Brandstetter, J., and Hochreiter, S. (2019). RUDDER: Return decomposition for delayed rewards. In *Advances in Neural Information Processing Systems (NIPS)*, (pp. 13566–13577).

Arras, L., Arjona-Medina, J., Widrich, M., Montavon, G., Gillhofer, M., Müller, K.-R., Hochreiter, S., and Samek, W. (2019a). Explaining and interpreting lstms. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, vol. 11700 of *Lecture Notes in Computer Science*, (pp. 211–238).

Arras, L., Horn, F., Montavon, G., Müller, K.-R., and Samek, W. (2017a). "What is relevant in a text document?": An interpretable machine learning approach. *PLoS ONE*, *12*(7), e0181142.

Arras, L., Montavon, G., Müller, K.-R., and Samek, W. (2017b). Explaining recurrent neural network predictions in sentiment analysis. In *Proceedings of the EMNLP'17 Workshop on Computational Approaches to Subjectivity, Sentiment & Social Media Analysis (WASSA)*, (pp. 159–168).

Arras, L., Osman, A., Müller, K.-R., and Samek, W. (2019b). Evaluating recurrent neural network explanations. In *Proceedings of the ACL'19 BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, (pp. 113–126).

Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., and Samek, W. (2015). On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS ONE*, *10*(7), e0130140.

Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., and Müller, K.-R. (2010). How to explain individual classification decisions. *Journal of Machine Learning Research*, *11*, 1803–1831.

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and trsanslate. *arXiv preprint arXiv:1409.0473*.

Baldi, P., Sadowski, P., and Whiteson, D. (2014). Searching for exotic particles in high-energy physics with deep learning. *Nature Communications*, *5*, 4308.

Balduzzi, D., Frean, M., Leary, L., Lewis, J., Ma, K. W.-D., and McWilliams, B. (2017). The shattered gradients problem: If resnets are the answer, then what is the question? In *34th International Conference on Machine Learning (ICML)*, (pp. 342–350).

Bau, D., Zhou, B., Khosla, A., Oliva, A., and Torralba, A. (2017). Network dissection: Quantifying interpretability of deep visual representations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 6541–6549).

Chang, C.-H., Creager, E., Goldenberg, A., and Duvenaud, D. (2018). Explaining image classifiers by counterfactual generation. *arXiv preprint arXiv:1807.08024*.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (pp. 1724–1734).

Cireşan, D., Meier, U., Masci, J., and Schmidhuber, J. (2011). A committee of neural networks for traffic sign classification. In *International Joint Conference on Neural Networks (IJCNN)*, (pp. 1918–1921).

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ding, Y., Liu, Y., Luan, H., and Sun, M. (2017). Visualizing and understanding neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, (pp. 1150–1159).

Dombrowski, A.-K., Alber, M., Anders, C., Ackermann, M., Müller, K.-R., and Kessel, P. (2019). Explanations can be manipulated and geometry is to blame. In *Advances in Neural Information Processing Systems*, (pp. 13567–13578).

Doshi-Velez, F., and Kim, B. (2017). Towards a rigorous science of interpretable machine learning. *abs/1702.08608*.

Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., and Thrun, S. (2017). Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, *542*(7639), 115.

Fong, R. C., and Vedaldi, A. (2017). Interpretable explanations of black boxes by meaningful perturbation. In *IEEE International Conference on Computer Vision (CVPR)*, (pp. 3429–3437).

Goodman, B., and Flaxman, S. (2017). European union regulations on algorithmic decision-making and a “right to explanation”. *AI Magazine*, *38*(3), 50–57.

Goyal, Y., Khot, T., Summers-Stay, D., Batra, D., and Parikh, D. (2017). Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., and Schmidhuber, J. (2017). LSTM: A Search Space Odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, *28*(10), 2222–2232.

Guyon, I., and Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, *3*(Mar), 1157–1182.

Hannun, A. Y., Rajpurkar, P., Haghpanahi, M., Tison, G. H., Bourn, C., Turakhia, M. P., and Ng, A. Y. (2019). Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. *Nature Medicine*, *25*(1), 65.

Hansen, K., Baehrens, D., Schroeter, T., Rupp, M., and Müller, K.-R. (2011). Visual interpretation of kernel-based prediction models. *Molecular Informatics*, *30*(9), 817–826.

Hochreiter, S., and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Horst, F., Lapuschkin, S., Samek, W., Müller, K.-R., and Schöllhorn, W. I. (2019). Explaining the unique nature of individual gait patterns with deep learning. *Scientific Reports*, *9*, 2391.

Johnson, J., Hariharan, B., van der Maaten, L., Fei-Fei, L., Zitnick, C. L., and Girshick, R. B. (2017a). CLEVR: A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 1988–1997).

Johnson, J., Hariharan, B., van der Maaten, L., Hoffman, J., Fei-Fei, L., Lawrence Zitnick, C., and Girshick, R. (2017b). Inferring and executing programs for visual reasoning. In *IEEE International Conference on Computer Vision (ICCV)*.

Jurmeister, P., Bockmayr, M., Seegerer, P., Bockmayr, T., Treue, D., Montavon, G., Vollbrecht, C., Arnold, A., Teichmann, D., Bressem, K., Schüller, U., von Laffert, M., Müller, K.-R., Capper, D., and Klauschen, F. (2019). Machine learning analysis of DNA methylation profiles distinguishes primary lung squamous cell carcinomas from head and neck metastases. *Science Translational Medicine*, *11*(509).

Kauffmann, J., Esders, M., Montavon, G., Samek, W., and Müller, K.-R. (2019). From clustering to cluster explanations via neural networks. *arXiv preprint arXiv:1906.07633*.

Kauffmann, J., Müller, K.-R., and Montavon, G. (2020). Towards explaining anomalies: a deep taylor decomposition of one-class models. *Pattern Recognition*, *101*, 107198.

Kim, B., Wattenberg, M., Gilmer, J., Cai, C., Wexler, J., Viegas, F., and Sayres, R. (2018). Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International Conference on Machine Learning*, (pp. 2668–2677). PMLR.

Kohlbrenner, M., Bauer, A., Nakajima, S., Binder, A., Samek, W., and Lapuschkin, S. (2020). Towards best practice in explaining neural network decisions with lrp. In *IEEE International Joint Conference on Neural Networks (IJCNN)*, (pp. 1–7).

Lapuschkin, S., Binder, A., Montavon, G., Müller, K.-R., and Samek, W. (2016a). Analyzing classifiers: Fisher vectors and deep neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 2912–2920).

Lapuschkin, S., Binder, A., Montavon, G., Müller, K.-R., and Samek, W. (2016b). The lrp toolbox for artificial neural networks. *The Journal of Machine Learning Research*, *17*(1), 3938–3942.

Lapuschkin, S., Wäldchen, S., Binder, A., Montavon, G., Samek, W., and Müller, K.-R. (2019). Unmasking clever hans predictors and assessing what machines really learn. *Nature Communications*, *10*, 1096.

Li, J., Monroe, W., and Jurafsky, D. (2016). Understanding Neural Networks through Representation Erasure. *arXiv preprint arXiv:1612.08220*.

Lu, C., and Tang, X. (2015). Surpassing human-level face verification performance on LFW with GaussianFace. In *29th AAAI Conference on Artificial Intelligence*, (pp. 3811–3819).

Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., and Lee, S.-I. (2020). From local explanations to global understanding with explainable ai for trees. *Nature Machine Intelligence*, *2*(1), 2522–5839.

Lundberg, S. M., and Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems (NIPS)*, (pp. 4765–4774).

Macdonald, J., Wäldchen, S., Hauch, S., and Kutyniok, G. (2019). A rate-distortion framework for explaining neural network decisions. *arXiv preprint arXiv:1905.11092*.

Mascharka, D., Tran, P., Soklaski, R., and Majumdar, A. (2018). Transparency by design: Closing the gap between performance and interpretability in visual reasoning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 4942–4950).

Mayr, A., Klambauer, G., Unterthiner, T., and Hochreiter, S. (2016). Deeptox: toxicity prediction using deep learning. *Frontiers in Environmental Science*, *3*, 80.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533.

Montavon, G. (2019). Gradient-based vs. propagation-based explanations: An axiomatic comparison. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, vol. 11700 of *Lecture Notes in Computer Science*, (pp. 253–265). Springer.

Montavon, G., Binder, A., Lapuschkin, S., Samek, W., and Müller, K.-R. (2019). Layer-wise relevance propagation: An overview. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, vol. 11700 of *Lecture Notes in Computer Science*, (pp. 193–209). Springer.

Montavon, G., Lapuschkin, S., Binder, A., Samek, W., and Müller, K.-R. (2017). Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, *65*, 211–222.

Montavon, G., Samek, W., and Müller, K.-R. (2018). Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, *73*, 1–15.

Moravčík, M., Schmid, M., Burch, N., Lisý, V., Morrill, D., Bard, N., et al. (2017). Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, *356*(6337), 508–513.

Osman, A., Arras, L., and Samek, W. (2020). Towards ground truth evaluation of visual explanations. *arXiv preprint arXiv:2003.07258*.

Poerner, N., Schütze, H., and Roth, B. (2018). Evaluating neural network explanation methods using hybrid documents and morphosyntactic agreement. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, (pp. 340–350).

Reyes, E., Estévez, P. A., Reyes, I., Cabrera-Vives, G., Huijse, P., Carrasco, R., and Forster, F. (2018). Enhanced rotational invariant convolutional neural network for supernovae detection. In *International Joint Conference on Neural Networks (IJCNN)*, (pp. 1–8).

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). Why should i trust you?: Explaining the predictions of any classifier. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, (pp. 1135–1144).

Robnik-Šikonja, M., and Bohanec, M. (2018). Perturbation-based explanations of prediction models. In *Human and Machine Learning*, (pp. 159–175). Springer.

Samek, W., Binder, A., Montavon, G., Lapuschkin, S., and Müller, K.-R. (2017). Evaluating the visualization of what a Deep Neural Network has learned. *IEEE Transactions on Neural Networks and Learning Systems*, *28*(11), 2660–2673.

Samek, W., Montavon, G., Lapuschkin, S., Anders, C. J., and Müller, K.-R. (2020). Toward interpretable machine learning: Transparent deep neural networks and beyond. *arXiv preprint arXiv:2003.07631*.

Samek, W., Montavon, G., Vedaldi, A., Hansen, L. K., and Müller, K.-R. (2019). *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. vol. 11700 of *Lecture Notes in Computer Science*. Springer.

Samek, W., and Müller, K.-R. (2019). Towards explainable artificial intelligence. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, vol. 11700 of *Lecture Notes in Computer Science*, (pp. 5–22). Springer.

Santoro, A., Raposo, D., Barrett, D. G., Malinowski, M., Pascanu, R., Battaglia, P., and Lillicrap, T. (2017). A Simple Neural Network Module for Relational Reasoning. In *Advances in Neural Information Processing Systems (NIPS)*, (pp. 4967–4976).

Schütt, K. T., Arbabzadah, F., Chmiela, S., Müller, K. R., and Tkatchenko, A. (2017). Quantum-chemical insights from deep tensor neural networks. *Nature Communications*, *8*, 13890.

Shrikumar, A., Greenside, P., Shcherbina, A., and Kundaje, A. (2016). Not just a black box: Learning important features through propagating activation differences. *arXiv preprint arXiv:1605.01713*.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., et al. (2016). Mastering the game of go with deep neural networks and tree search. *Nature*, *529*(7587), 484–489.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. (2017). Mastering the game of Go without human knowledge. *Nature*, *550*(7676), 354–359.

Simonyan, K., Vedaldi, A., and Zisserman, A. (2013). Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

Simonyan, K., and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Smilkov, D., Thorat, N., Kim, B., Viégas, F., and Wattenberg, M. (2017). Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*.

Sundararajan, M., Taly, A., and Yan, Q. (2017). Axiomatic attribution for deep networks. In *International Conference on Machine Learning (ICML)*, (pp. 3319–3328).

Swartout, W. R., and Moore, J. D. (1993). Explanation in second generation expert systems. In *Second Generation Expert Systems*, (pp. 543–585). Springer Berlin Heidelberg.

Thomas, A. W., Heekeren, H. R., Müller, K.-R., and Samek, W. (2019). Analyzing neuroimaging data through recurrent deep learning models. *Frontiers in Neuroscience*, *13*, 1321.

von Lilienfeld, O. A., Müller, K.-R., and Tkatchenko, A. (2020). Exploring chemical compound space with quantum-based machine learning. *Nat. Rev. Chem.*, *4*, 347—-358.

Warnecke, A., Arp, D., Wressnegger, C., and Rieck, K. (2020). Evaluating Explanation Methods for Deep Learning in Security. In *Proceedings of the 2020 IEEE European Symposium on Security and Privacy*, (pp. 158–174).

Weller, A. (2019). Transparency: Motivations and challenges. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, vol. 11700 of *Lecture Notes in Computer Science*, (pp. 23–40). Springer.

Yang, Y., Tresp, V., Wunderle, M., and Fasching, P. A. (2018). Explaining Therapy Predictions with Layer-Wise Relevance Propagation in Neural Networks. In *IEEE International Conference on Healthcare Informatics (ICHI)*, (pp. 152–162).

Yang, Z., He, X., Gao, J., Deng, L., and Smola, A. J. (2016). Stacked attention networks for image question answering. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 21–29).

Yeom, S.-K., Seegerer, P., Lapuschkin, S., Wiedemann, S., Müller, K.-R., and Samek, W. (2019). Pruning by explaining: A novel criterion for deep neural network pruning. *arXiv preprint arXiv:1912.08881*.

Zeiler, M. D., and Fergus, R. (2014). Visualizing and Understanding Convolutional Networks. In *Computer Vision - ECCV 2014*, (pp. 818–833).

Zhang, J., Lin, Z. L., Brandt, J., Shen, X., and Sclaroff, S. (2016). Top-down neural attention by excitation backprop. In *Proceedings of the 14th European Conference on Computer Vision (ECCV)*, (pp. 543–559).

Zintgraf, L. M., Cohen, T. S., Adel, T., and Welling, M. (2017). Visualizing deep neural network decisions: Prediction difference analysis. *International Conference on Learning Representations (ICLR)*.