Simplifying the explanation of deep neural networks with sufficient and necessary feature-sets: case of text classification

Florentin Flambeau Jiechieu Kameni[a,b,*], Norbert Tsopze[a,b]

[a]*Department of Computer Sciences - University of Yaounde I, Yaounde, Cameroon*
[b]*IRD, UMIMISCO, F-93143 Bondy, France*

Abstract

During the last decade, deep neural networks (DNN) have demonstrated impressive performances solving a wide range of problems in various domains such as medicine, finance, law, etc. Despite their great performances, they have long been considered as black-box systems, providing good results without being able to explain them. However, the inability to explain a system decision presents a serious risk in critical domains such as medicine where people's lives are at stake. Several works have been done to uncover the inner reasoning of deep neural networks. Saliency methods explain model decisions by assigning weights to input features that reflect their contribution to the classifier decision. However, not all features are necessary to explain a model decision. In practice, classifiers might strongly rely on a subset of features that might be sufficient to explain a particular decision. The aim of this article is to propose a method to simplify the prediction explanation of One-Dimensional (1D) Convolutional Neural Networks (CNN) by identifying sufficient and necessary features-sets. We also propose an adaptation of Layer-wise Relevance Propagation for 1D-CNN. Experiments carried out on multiple datasets show that the distribution of relevance among features is similar to that obtained with a well known state of the art model. Moreover, the sufficient and necessary features extracted perceptually appear convincing to humans.

---

*Corresponding author
Email addresses: `florentin.jiechieu@facsciences-uy1.cm` (Florentin Flambeau Jiechieu Kameni), `florentin.jiechieu@mintp.cm` (Florentin Flambeau Jiechieu Kameni), `norbert.tsopze@facsciences-uy1.cm` (Norbert Tsopze)

1. Introduction

In human understanding, a decision should be explained. Medical practitioners use diagnostic results to explain medical prescriptions to their patients, while lawyers explain court decisions using facts. Artificial intelligence models have evolved over the years, becoming more and more sophisticated and providing outstanding results in solving complex problems in a wide range of applications such as : text processing, image processing, medical analysis, etc. In the beginning, researchers were just concerned about improving the accuracy of artificial neural models. Therefore, deep neural networks (DNN) have emerged providing better results than the past models. Even though the results provided by DNN are satisfactory enough, they still suffer from the problem of opacity. They are often considered as black boxes, providing results without being able to explain them. In fact, entrusting important decisions to a system that cannot be explained presents obvious risks [1].

The problem of model interpretability has become a major concern in machine learning research communities since some machine learning programs were reported being racist, making social discrimination despite having good results in tests data [2, 3]; other programs were found using inappropriate features (background of an image for example) to compute their decisions [2, 4].

Model explanation presents more than one benefit, among them : it eases the acceptability of artificial models in critical domains such as medicine, finance, etc.; it provides insights to improve learning algorithms and thus, models accuracy. Models explanation can also provide knowledge in the absence of domain theory.

Techniques to explain artificial models vary from one kind of model to another. While it is easy to interpret a decision tree or a model based on association rules for instance, the explanation of deep neural networks can be challenging. Several techniques reported in [5, 2, 4, 1] have been proposed to explain deep neural networks :

1. Visualizing the network parameters: In images classification this method consists in visualizing the convolution filters and projecting them on the input images to highlight the regions of the images used for the classification

2. Inspecting the model: This method consists in providing a set of representations to understand how the model works and why it returns certain predictions more than others.
3. Using a saliency mask: this method consists in pointing out the cause of certain model outputs. It is very important in image or text classification because users can visualize the part of the input mainly responsible for the model outcome.
4. The sensibility analysis which consists in finding the influence of a given input feature to the model outcome.

The present paper focuses on the local interpretation of CNN models built for text classification, which is the ability to explain or provide a meaningful and understandable justification to the model decision for a given input. The majority of works on CNN models explanation concerns image processing [5, 2]. Only few research works have been conducted to interpret CNN predictions built for text classification [6, 7, 8, 9].

Previews studies have focused on the visualization of n-grams detected by filters. Thereafter, Jacovi et al. [6] showed that some convolutional filters select accidental n-grams features because no other n-grams scored higher than them and those n-grams are not relevant in the explanation. However, even though certain n-grams would be considered accidental, they still influence the classifier decision. Moreover, their method lacks of genericity because it only applies to CNN with no hidden layer in the fully connected stage.

In this article, we propose a method to explain CNN built for text classification using the principle of Layer-wise Relevance Propagation (LRP) [10], with a small adaptation of the contribution ratio formula which enables to clearly distinguish positive contributing n-gram features from negative ones.

Indeed, positive contributing n-gram features are n-grams that contribute to increase the value of the predicted class and negative n-grams have the opposite effect. Therefore, positive n-grams are likely the features that might be useful to explain a classifier decision. However, as Miller suggested, not all features are necessary to explain a model decision [4].

Carter et al. [11] have recently proposed to find the collection of Sufficient Input Subsets (**SIS**) which they defined as the minimal subset of features whose observed values alone suffice for the same decision to be reached, even if all other input feature values are missing. However, they do not distinguish positive features from negative ones. Indeed, the presence of only positive features without any negative ones might naturally suffice for the same decision to be

reached. But in practice, inputs generally consists of both positive and negative features. That is why we think that sufficient features must absolutely be determined in presence of all negative features. In that case, sufficient features would be subsets of positive features that in presence of negative ones suffice for the same decision to be reached.

In this study we refine the definition proposed by [11] and propose the concept of Sufficient feature-sets which are subsets of positive contributing features such that the inhibition of the effect of all the other positive features does not change the classifier decision. We also propose to find necessary features which are positive features such that their inhibition changes the classifier decision.

Indeed, necessary features if exists, represent key features (key n-grams responsible for the decision). On the other hand, sufficient feature-set can be used to simplify the explanation provided to human in the light that not all positive contributing features are not necessary to explain the model decision.

The main contributions of this work are :

1. We propose an LRP-based method to determine the relevance of n-grams features in a 1D-CNN architecture built for text classification ;
2. We also propose algorithms to evaluate sufficient and necessary feature sets in order to simplify the explanation, highlighting key features ;
3. We carried out multiple experiments in different applications of text classification including sentiment analysis and question answering to demonstrate the effectiveness our the method.

The rest of this article is structured as follows: section 2 focuses on the presentation of some closely related works. Section 3 presents the principle of text classification using CNN; section 4 summarizes the LRP method. In section 5, our method to explain 1D-CNN is discussed, algorithms used to compute sufficient feature-sets and necessary features are also presented. The experiments and results are the aim of the sixth section.

2. Related Works

The explanation of deep learning models often presented as black box models has become a topical issue in recent years. Most of the works about model explanation reported in these surveys [12, 2] are concerned with image processing and they cannot easily be transposed to text processing due to the discrete nature of texts [6]. Works aiming to interpret machine learning algorithm

can be grouped in two major categories according to [12]: interpretability by mathematical structures which concerns the use of these structures to reveal the mechanism of machine learning and neural network algorithms [13, 14, 15, 16, 17, 18]; and the perceptive algorithm which tends to present the user with explanatory elements that can be humanly perceived. The perceptive explainability includes : the saliency method, which explains the decision of an algorithm by assigning values that reflect the importance of input components based on their contribution to the classifier decision [9, 19, 20, 21, 22, 23]; the signal method that observes the stimulation of neurons or a collection of neurons [24, 25, 26]; and the verbal method in which verbal chunks or sentences are provided to the user as explanation [27, 27, 28]. In this section, we will focus on the saliency method as the explanation method proposed in this article falls into this category.

LIME (Local Interpretable Model-Agnostic Explanations) [7] is a model independent interpretability method, in the sense that it is able to explain without needing to explore the inner functioning of the model. The principle of LIME method is to perturb the input around its neighborhood and observe the changes in the model predictions. For example, suppose we want to explain the prediction for the sentence "I hate this movie". Lime will perturb that sentence and get the output predictions of resulting sentences such as : "I this movie", "hate this movie", "hate movie", etc. From these perturbations, the relevancy of input words are determined. Even though LIME produces good results in practice, it does not "unblackbox" the model and the fidelity of the explanation to the exact inner model functioning can still be questioned. Another limit of LIME is that, to explain a single instance, it computes the outcome of multiple other instances obtained by varying the initial instance : this introduces a supplementary cost in terms of time complexity.

Jacovi et al. [6] have developed a method to understand how the CNN classifies a text. They examined the CNN parameters and showed that filters used in the convolution layer may capture several different semantic classes of n-gram by using different activation patterns : Informative n-grams selected by the pooling and used to classify the text, and, Uninformative n-gram eliminated by the pooling. They also distinguish between n-grams : deliberate and accidental n-grams. Deliberate n-grams are effectively informative with higher score regarding the final decision, accidental n-gram are n-gram selected by the pooling despite having a low score, because no other n-gram feature scored higher than them. However, their method applies only to a limited range of CNN architectures, those with only one layer in the fully connected stage. In

addition, the fact that different convolution filters may select the same n-gram is not taken into account when scoring the n-gram features.

The Layer-wise relevance propagation was firstly introduced in [10]. In this method, the relevance of input features is evaluated by computing their contributions to the output of a particular neuron. The contributions are evaluated such that, the sum of contributions of units in a layer to a unit in the next layer is equal to the output of the latter. Other articles using this method includes [29, 29, 30, 31]. In [31] authors use LRP to explain CNN as for images. LRP is used to weight each component of the word embedding vectors as if they were pixels. The relevance of a word is obtained by summing up the relevance of each component of the corresponding word vector.

3. Text classification using CNN

The architecture of the CNN described in this section is based on the architecture proposed by Kim [32] for sentence classification using CNN. Below we explain how the components of the CNN architecture work. Figure 1 shows the different layers of the architecture. Each layer realizes specific operations.
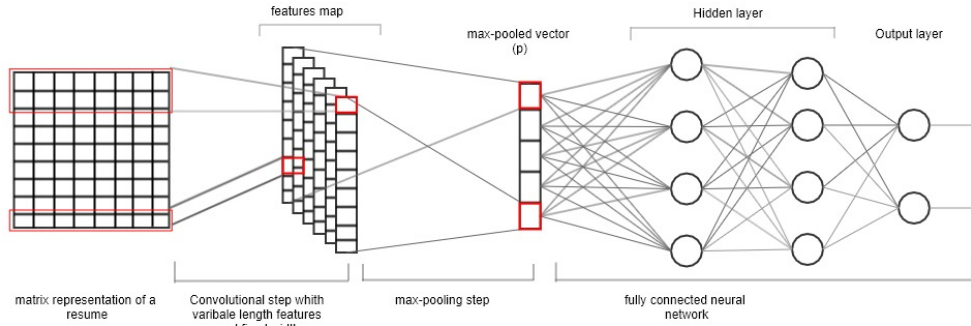

Figure 1: Text classification using CNN

The CNN Input layer. The CNN takes as input a sequence of word embedding vectors representing the text to classify. A word w can be modeled as a d-dimension vector of numerics : $w \in R^d$. Consequently, a text of length n which is a sequence of n words can be modeled as a matrix M of dimension $n \times d$ ie $M = w_1, w2, ..., w_n \in R^{n \times d}$. An l-word n-gram $u_i = < w_i, .., w_{i+l-1} >$ $(0 \le i \le n - l)$ is a sequence of l consecutive words in the input text, $u_i \in R^{l \times d}$.

1D-Convolution layer. The convolution layer consists of 1D-convolution filters. A 1-kernel size convolution filter f_j can be modeled as a $d-dimension$ vector (same shape as a word). The kernel size of a filter refers to the height of the filter sliding window since the width is fixed and corresponds to the word embedding dimension. A filter of kernel size 2 will slide vertically over two consecutive words at one time as shown in Figure 1. The convolution performs the scalar product $< u_i, f_j >$ between an n-gram u_i and a filter f_j. The convolutions of a particular filter f_j among the whole input matrix results in a column vector $F_{,j}$ called features map associated to f_j, and for the whole filters, the result is a matrix $F \in R^{(n-l+1) \times m}$ where m is the total number of filters and the columns of the matrix represent single features maps associated to convolution filters. The values inside the features map are rectified by setting all negative values to zero (Rectified Linear Unit) before going as input to the max-pooling layer. To summarize, the convolution layer can be modeled as a multivariate function $c : R^{n \times d} \rightarrow R^{(n-l+1) \times m}$ which takes as input a matrix of words embedding and returns a matrix of feature maps.

Max-pooling layer : Global Max-pooling. The global max-pooling filter picks the maximum value in each rectified features map (each column of F), that is the value corresponding to the word which had the highest convolution score with the filter associated to that features map. The result of the global max-pooling is a $m-dimension$ vector. The max-pooling layer can then be modeled as function $mp : R^{(n-l+1) \times m} \rightarrow R^m$ which takes a matrix of features map and returns a vector containing the maximum value in each column.

Fully connected neural network (FCNN). The FCNN takes as input the max-pooled vector P (vector resulting from the max-pooling operation) and produces an output vector corresponding to the activation of each output unit. Therefore, the FCNN can be modeled as a function $h : R^m \rightarrow R^c$, c being the number of output units. Let's denote by L the total number of layers in the FCNN, h^i the output vector of the $i\text{-}th$ layer, starting from $h^0 = P$ (the max-pooled vector), and finishing with a special output layer h^L which computes the output of the network. If g is the activation function of hidden units, then, the activations of units in layer k (in matrix-vector notation) is given by the equation 1 where b^k is the vector of biases and W^k is the matrix of weights connecting layer $k-1$ to layer k.

$$h^k = g(W^k h^{k-1} + b^k) \tag{1}$$

7

The activation of a single unit i in layer k is given by the Eq. 2

$$h_i^k = g(\sum_j W_{ij}^k h_j^{k-1} + b_i^k) \tag{2}$$

Then the activations of the output units will be given by the Eq. 3 (in matrix-vector notation):

$$h^L = g(W^L h^{L-1} + b^L) \tag{3}$$

The 1D-CNN mathematically can be modeled as a function f which realizes the composition of the functions c, m and h : $f = h \circ m \circ c$.

4. Layer-wise relevance propagation (LRP)

LRP [10] is a technique used to evaluate the contribution of input features to the neural network's output. The principle is to compute the contributions of units in the last hidden layer, then back-propagate them up to the input features. In this section, we will show how LRP is used to evaluate the contribution of the component of the max-pooled vector (see Figure 1).

Let's denote by $f : R^l \rightarrow R^c$ the vector-value multivariate representing the mapping between the inputs features and the output of the network, where l represents the size of the input vector (size of the max-pooled vector) and c represents the number of output classes. We denote by $z_{ij}^l = w_{ij}^{(l+1)} h_j^{(l)}$ the product of the activation of the unit i in layer l and the synaptic weight connecting i to the unit j in layer $l + 1$.

We denote by $z_j^l = \sum_i w_{ij}^l h_i^{(l-1)} + b_j^l = \sum_i z_{ij}^l + b_j^l$ ($0 <= l < L$) the preactivation of the neuron j in layer l.

Definition 1. *The contribution of a unit i of the $l - th$ layer to a unit j of the $(l+1)^{th}$ layer ($0 <= l < L$) denoted by $R_{ij}^{(l,l+1)}$ is the extend to which the unit i has contributed to the preactivation value of the unit j.*

Formally the contribution of the unit i to the output unit j is calculated using the equation 4 where w_{ij}^l is the connection weight between the neuron i of the layer l and the unit j of the layer $k + 1$; and h_i^l the activation of the unit i.

$$R_{ij}^{(l,l+1)} = \frac{z_{ij}^l}{z_j^{(l+1)}} = \frac{h_i^l * w_{ij}^{(l+1)}}{\sum_k h_k^l w_{kj}^{(l+1)} + b_j^{(l+1)}} \tag{4}$$

8

To prevent $R_{ij}^{(l,l+1)}$ to take underbounded values due to potential small values of $z^{(l+1)} + 1$ we will consider equation 5 instead, where ϵ is the stabilizer.

$$R_{ij}^{(l,l+1)} = \begin{cases} \dfrac{z_{ij}^l}{z_j^{(l+1)} + \epsilon} & \text{if } z_j^{(l+1)} \geq 0 \\[2em] \dfrac{z_{ij}^l}{z_j^{(l+1)} - \epsilon} & \text{if } z_j^{(l+1)} \leq 0. \end{cases} \tag{5}$$

Knowing the contribution of units of the $(l+1) - th$ layer ($0 \leq l < k - 1$) to the value predicted by the output neuron j, we can evaluate the contribution R_{ij}^l of each neuron i in layer l to the value predicted by the output neuron j by the Eq. 6.

$$R_{ij}^l = \sum_k R_{ik}^{(l,l+1)} * R_{kj}^l \tag{6}$$

where $R_{jj}^L = f_j(x)$ and R_{ij}^0 represents the contribution of input feature x_i to the output neuron j.

The mathematical foundations of LRP method are discussed in [10]. The LRP algorithm operates as follows :

We start by computing the contribution of each unit i in the (L-1)th layer to the value predicted for each unit j in the output layer (L-th layer) using the equation 7

$$R_{ij}^{L-1} = R_{ij}^{(L-1,L)} * f_j(x) \tag{7}$$

$R^{(L-1,L)}$ is calculated using equation 5.

Thereafter, the contributions of units in layer $L - 2$ to layer $L - 1$ are calculated using the recurrent equation 6. The process is iterated until the contribution of input features to the outputs values predicted by the network are determined.

LRP has widely been applied to image processing in order to weigh input pixels. But the discrete nature of text makes it difficult to apply in text CNN without any adaptation.

5. Prediction Explanation

The goal of the prediction explanation here is to highlight key features which have led to the classifier decision and that can be perceived by a human as a

9

reason for that decision. As described in Sect. 3, each component of the max-pooled vector corresponds to the maximum convolution results of a particular filter with each words' vector or sequences of words' vectors of the input text depending on the filter kernel size used. Therefore, recovering the n-gram selected by a filter of kernel size k simply consists in selecting the sequence of k-words which produced the maximum convolution result with that filter. Only those n-grams will further influence the classifier final decision [6].

Based on the later assumption, the general principle of the explanation method described in this section consists in : (1) using the LRP approach to compute the relevance of each component of the max-pooled vector; (2) then recovering the n-grams associated to each of those components and determine their relevance. The method to explain the model prediction for an input text can be recapitulated as follows :

1. First, a text is presented as input to the CNN and the outputs $f(x)$ is determined (Classification);
2. Next, an adapted LRP that we will call LRP-A is used to compute the contribution of each component of the max-pooled vector to the output $f(x)$;
3. Then, the n-gram features are determined and their relevance computed;
4. After, sufficient and necessary n-gram features are computed;
5. Finally, n-grams are classified into positive, negative sufficient or necessary.

The first step is described in detail in section 3. This section will focus on the prediction explanation method.

5.1. LRP ratio Adaptation

Sect. 4 describes the the LRP method and its variants (LRP-ϵ, LRP-0, LRP-$\alpha\beta$). However, we found a problem with either of these methods.

Let's consider the simplified neural network presented in Figure 2. Generally, activation functions in neural networks are increasing functions. This means that the greater the input of the function, the greater its output value. However, based on the architecture presented in Figure 2, the contribution ratio of the input unit a_1 to the linear sum of product z will be $a_1 w_1 / \sum a_i w_i = -2/-1 = 2$, and the contribution of a_2, a_3 and a_4 will respectively be -2, -1, and 1. This means that a_1 will be considered as the highest contributing input feature though it has a negative effect on the sum z because it contributes to reduce the value of z, thus reducing the activation of the output unit. On the other hand, a_2
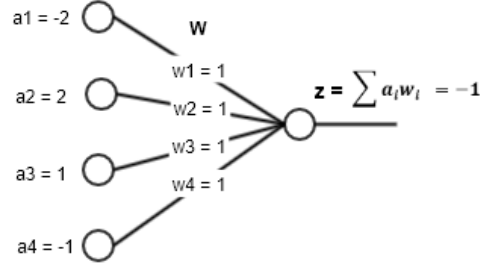
10

Figure 2: Simplified neural network

which actually contributes to increase the value of the sum of product z will be considered having a negative contribution. Situations where the sum z is negative, could distort the interpretation of some computed relevances when using Eq. 4. To avoid that behavior, the contribution ratio of a unit i is calculated by dividing the product $a_i w_i$ by the sum of absolute values of the products $z' = \sum |a_j w_j|$, which leads to Eq. 8. Using Eq. 8, the contributions of a_1, a_2, a_3 and a_4 will respectively be -0.33, 0.33, 0.16 and -0.16; which better reflects the effect of each input variable on the output unit pre-activation function.

$$R_{ij}^{(l,l+1)} = \frac{z_{ij}^l}{z_j^{(l+1)}} = \frac{a_i * w_i}{\sum_k |a_k w_k|} \tag{8}$$

5.2. Contribution of n-gram features

Having the output of an input x, the first step in the explanation is to calculate the contribution of each component of the max-pooled vector using the LRP (Sect. 4) method. We end-up with a matrix representing the contribution of each filter to each class. After that, the next step is to compute the contribution of each n-gram to the output units.

We recall that each component of the max-pooled vector corresponds to the maximum convolution value of the filter with every word vector of the embedding input matrix. A filter f_j selects an n-gram u_i if the convolution of that filter with the given n-gram has produced the maximum value. In addition, different filters may select the same n-gram for the same input text. Therefore, the contribution or the relevance of an n-gram u_i noted C_{u_i} is calculated as the sum of the contributions of filters (R_f) which select that n-gram (Eq. 9) .

$$R_{u_i} = \sum_{f_j \in A_i} R_{f_j} \tag{9}$$

11

R_{f_j} is the relevance of the max-pooled value associated to the filter f_j which also defines the relevance of that filter. R_{f_j} is calculated using the LRP method described in Sect. 4 A_i is the adjacency list associated to the n-gram u_i, which is the set of filters that selects the n-gram u_i. Formally, the n-gram selected by a filter f_j is the n-gram u_i^* which maximizes the scalar product $< u_i, f_j >$ ($1 \leq i \leq n - l - 1$, $1 \leq j \leq m$). where n is the number of words in the input text, l the number of words of the n-gram and m the total number of filters. u_i^* is determined by the formula 10 [33].

$$u_{i*} = \arg\max_u (< u_i, f_j >), u_i \in W \tag{10}$$

With (Eq. 9) it is possible to evaluate the contribution of each n-gram to each output independently. We could then consider that : the greater the value of the contribution of an n-gram to the value predicted for an output neuron j, the more it is relevant to explain the output of that neuron. However, when using softmax activation in the output layer, the activation of a unit depends on the output values of the other units. In that case, the relevance of an n-gram for a target class must also take into account its contribution to other classes. Therefore it would be fair to refine the latter assumption to : a relevant n-gram to a given class c_j is an n-gram which highly contributes to the target class c_j and contributes little to other classes. Based on this assumption, the relevance of an n-gram u_i with respect to an output class c_j is the difference between the contribution of that n-gram to the class c_j and the mean of its contributions to other classes. This means that : the more an n-gram contributes to the class c_j and the less it contributes to the other classes, the more it is relevant to the target class c_j.

$$R_j^{'i} = R_j^i - \frac{\sum_{p \neq j} R_p^i}{k - 1} \tag{11}$$

k is the number of classes. For example, table 1 shows the contributions of

Table 1: Contribution of some n-grams to the sentiment predicted for the sentence : *"great pocket pc phone combination"*

	NEG	POS	REL-POS
great	-0.0971	0.0822	0.1794
great pocket pc	-0.1132	0.0789	0.1922
phone combination	0.0019	0.0014	-0.0005

some n-grams to the sentiment predicted by a sentiment analysis model for the sentence *"great pocket pc phone combination"*. The model predicted the "positive" class (**POS**). From this table, we can observe that the contribution of the 3-gram *"great pocket pc"* to the positive class (**POS**) was less than that of the 1-gram *"great"*. However, we will consider the 3-gram *"great pocket pc"* as more relevant to the "positive" class because it also strongly negatively contributes to the "negative" class (**NEG**), compared to the word *great*.

For logistic output units, where the activation of a unit only depends on its output value we can consider that $R_j^{'i} = R_j^i$.

5.3. n-gram polarity

Definition 2. *Let u_i be the n-gram identified by the filter i and R^i the contribution vector of u_i to the output units, ie R_j^i is the contribution of u_i to the output class c_j. The class explained by u_i is the class c_j^* where u_i contributes the most. Formally,*

$$c_{j*} = \underset{j}{\arg\max}(R_j^i), 2 \leq j \leq k \tag{12}$$

k being the total number of classes.

Definition 3. *A positive n-gram for a target class c_j is an n-gram which mostly contributes to c_j among all other classes.*

An n-gram with negative contribution will be called a negative n-gram; and an n-gram with null contribution does not influence in the outcome of the target output unit.

Positive n-grams for a predicted class are n-grams which can likely be used to explain the prediction of the classifier because they actually contribute to increase the output value of the predicted class. But not all positive n-grams are necessary to explain the target class. We assume that only a subset of n-gram features is sufficient to explain a prediction. They are key n-gram features on which the model mainly base to compute its decision.

5.4. Sufficient features-set

let $X = (x_1, x_2, ..., x_n)$ an input data and $f(X) = y$. Let $U = \{u_i\}_{i=1}^{i=n}$ the set of n-gram features selected by convolution filters, $U^+ \subset U$ and $U^- \subset U$ such that $U = U^+ \cup U^-$, the subsets of positive respectively negative n-gram features selected by convolution filters with regards to the classifier output decision.

Definition 4. *A sufficient feature set is a subset $S \subset U^+$ of positive n-gram features such that the inhibition of all other positive features except S-features does not change the final classifier decision for an input text x. Inhibiting an n-gram feature consists in inhibiting all the filters that select this n-gram; and inhibiting a filter consists in setting the corresponding max-pooled value to zero or setting the synaptic weights connecting the max-pooled value associated to that filter to zero.*

A sufficient set of features S is minimum if there is no other sufficient set of features S' such that $S' \subset S$.

Definition 5. *The relevance of a sufficient feature-set S with respect to a target class c, is the sum of the contributions of the n-gram features of S to class c.*

We are interested in sufficient features-set with maximum relevance. Therefore, a greedy-like method can be used to find a minimum sufficient features-set with highest relevance. The aim of finding sufficient feature-sets is to simplify the explanation by providing to the user key features on which the model mainly relies on to produce its outcome. The algorithm 1 is a greedy-like algorithm to determine sufficient features set.

5.5. Necessary features

let $X = (x_1, x_2, ..., x_n)$ an input data and $f(X) = y$. Let $U = \{u_i\}_{i=1}^{i=n}$ the set of n-gram features selected by convolution filters, $U^+ \subset U$, $U^- \subset U$ such that $U = U^+ + U^-$, the subset of positive respectively negative n-gram features selected by convolution filters with regards to the classifier output decision.

Definition 6. *A necessary n-gram feature is a positive n-gram $u \in U^+$ such that the inhibition of filters that select u will change the classifier decision for the input text X.*

The necessary feature-set is the set of necessary features. Necessary features present a major importance in explanation because according to the model, they were required to produce the decision given by the model. Algorithm 2 is a greedy-like algorithm to determine the set of necessary features.

Algorithm 1 SF-set : Get Sufficient features-set

Require: A 1D-CNN model f ; An n-words input sequence X;

Ensure: S = A minimum sufficient features-set with maximum relevance

1: $p = f(X)$ ▷ The class predicted for the input X

2: $C = LRP(f, X)$ ▷ The contributions of input features of the fully connected neural network to the output units

3: $F = f.Conv1D(X)$ ▷ The intermediate result (Features map) of the 1D-convolution layer

4: $U = \arg\max_j F_{ij}$ ▷ Indices of n-gram features selected by convolution filters

5: $A = []$ ▷ Dictionary containing the list of filters that select each n-gram feature

6: NC = vector of n-gram contributions, initialized to null.

7: **for** i from 1 to m **do**

8:     $A[U[i]] = A[U[i]] + \{i\}$ ▷ add filter i into the adjacency list related to n-gram U[i]

9:     $NC[U[i]] = NC[U[i]] + C[i]$ ▷ C[i] is the contribution of filter i

10: **end for**

11: **if** the output activation of f is logistic **then**

12:     $U^+ = \{u \in U \mid C_p^i > 0\}$ ▷ p is the indice of the predicted class

13: **else if** output activation of f is softmax **then**

14:     $U^+ = \{u \in U \mid C_p^i - \frac{\sum_{j \neq p} C_j^i}{|C^i| - 1} > 0\}$

15: **end if**

16: $sort\ U^+\ elements\ in\ ascending\ order\ of\ their\ contribution$

17: $y = p,\ i = 0,\ S = U^+$

18: **while** p == y **do**

19:     $L = A[U[i]]$ ▷ List of filters selecting U[i]

20:     $f' =$ the model obtained by setting in f the weights connecting the max-pooled value corresponding to filters in L to zero

21:     $y = f'(X)$

22:     **if** y == P **then**

23:        Remove $U[i]$ from S ▷ U[i] is not necessary

24:     **end if**

25:     $i = i + 1$

26: **end while**

27: **return** S

Algorithm 2 NF-set: Get Necessary features-set

Require: A 1D-CNN model f; an n-words input sequence X;
Ensure: N = a set of necessary n-gram features
1: $p = f(X)$ ▷ The class predicted for input X
2: $C = LRP(f, X)$ ▷ The contributions of input features of the fully connected neural network to the output units
3: $F = f.Conv1D(X)$ ▷ The intermediate result (Features map) of the 1D-convolution layer
4: $U = \arg\max_j F_{ij}$ ▷ Indices of n-gram features selected by convolution filters
5: $A = []$ ▷ Dictionary containing the list of filters that select each n-gram feature
6: NC = vector of n-gram contributions, initialized to null.
7: **for** i from 1 to m **do**
8:       $A[U[i]] = A[U[i]] + \{i\}$
9:       $NC[U[i]] = NC[U[i]] + C[i]$
10: **end for**
11: **if** output activation of f is logistic **then**
12:       $U^+ = \{u \in U \mid C_p^i > 0\}$ ▷ p is the indice of the predicted class
13: **else if** output activation of f is softmax **then**
14:       $U^+ = \{u \in U \mid C_p^i - \frac{\sum_{j \neq p} C_j^i}{|C^i|-1} > 0\}$
15: **end if**
16: *sort U^+ elements in descending order of their contribution*
17: $y = p, \; i = 0, \; N = \{\}$
18: **while** $p \neq y$ **do**
19:       $L = A[U[i]]$
20:       $f' =$ the model obtained by setting in f the weights connecting the max-pooled value corresponding to filters in L of the FCNN input vector to zero
21:       $y = f'(X)$
22:       **if** $y \neq p$ **then**
23:           $N = N + U[i$ ▷ U[i] is necessary
24:       **end if**
25:       $i = i + 1$
26: **end while**
27: **return** N

---

6. Experiments and Results

The experiments[1] have been conducted on multiple channel CNN models based on Kim architecture [32] plus a trainable embedding layer. The explanation method has been tested on architectures consisting of 1 to 3 channels with filters of kernel sizes 1, 2 or 3. The word embedding layer consists of 50-dimension word vectors and the input texts were padded to a maximum length equals to 50.

6.1. Datasets

The method has been experimented on 4 datasets :

- **IMDB :** A movie review dataset for binary sentiment classification [34];

- **sentiment140 :** A dataset containing 1,600,000 tweets extracted using the twitter api for the sentiment analysis task [35];

- **TREC-QA_5500 :** A dataset for Question Answering Track with 5500 train samples [36]

- **TREC-QA_1000 :** A dataset for Question Answering Track with 1000 train samples [36];

- Few already pretraited training data from **Imdb**, **Amazon** and **yelp** were also provided by [37].

Table 2 describes the classes of the above mentioned datasets.

A model was built for each task described above. Table 3 shows a summary description of models used to test the explanation method described in this article. The model name is related to the name of the dataset used to train the model.

Table 3 shows that the model built for sentiment classification using **Imdb** dataset has 3 channels with filters of kernel size respectively of 1, 2, 3 and each of these channels has 40 filters; this makes a total of 120 filters in the convolution layer.

Table 2: Dataset classes description

dataset name	number of classes	Classes description
IMDB	2	0 = Negative 1 = positive
sent140	3	0 = Negative 2 = Neutral 4 = Positive
TREC-QA	6	DESC=DESCRIPTION ENTY=ENTITY HUM=HUMAN NUM=NUMBER LOC=LOCATION ABBR=ABBREVIATION

6.2. Results and Discussions

The explanation method was tested on models described in table 3. Figure 3 shows the distribution of relevance among positive n-grams for random sentences picked from the datasets described in Sect. 6.1. Labels with asterix (*) represent sufficient features sets detected by algorithm 1. They are also highlighted in red in the original sentence. Necessary features are terms inside the brackets. In Figure 3c the model classified the question according to the type of the expected answer. Indeed, the sentence *"how far is it from Denver to Aspen?"* is classified as **"NUM"** because the expected answer is a number representing the distance between Denver and Aspen. Several positive contributing n-grams *(far, denver, from denver, ...)* were identified, but only the n-grams *"far is"* and *"how far is"* were sufficient to produce the same decision. This means that if all the filters that select other positive n-grams were inhibited except those selecting *"far is"* and *"how far is"*, the decision would have been the same. Likewise, Figure 3d has multiple positive n-gram features, however only three n-gram features contributed enough such that they were sufficient to produce the final decision according to the model. They are : *"best", "best scene", "best scene in"*;

---

[1]https://github.com/florex/xai-cnn-lrp

Table 3: Description of models built to test the explanation method

| model name	number of channels	kernel sizes per channel	filters per channel	model accuracy
IMDB	3	[1,2,3]	[40,40,40]	83.6%
sent140	3	[1,2,3]	[40,40,40]	88,9%
TREC-QA_5500_1ch	1	[1]	[40]	78.14%
TREC-QA_5500_3ch	3	[1,2,3]	[40,40,40]	84.52%
TREC-QA_1000_3ch	3	[1,2,3]	[40,40,40]	74.80%

and when projected on the sentence it gives *"the best scene in the movie was when gerardo is..."*, which greatly simplifies human perception of the explanation provided.

In Figure 3a, the n-gram *"not good"* is at the same time sufficient and necessary, which means that it has greatly contributed to classify the sentence *soggy and [not good]* as *NEGATIVE*.

Table 4: Relevance of n-grams to the class predicted for the question : *"who was the star witness at the senate watergate hearings?"*

|	DESC	ENTY	ABBR	HUM	NUM	LOC	REL-HUM
senate	0.02	-0.02	-0.00	-0.00	0.01	-0.00	-0.00
witeness	0.01	-0.04	-0.01	-0.00	0.04	-0.01	-0.00
hearings	0.03	0.03	0.02	0.00	-0.03	-0.03	0.01
water-gate	0.01	0.01	0.01	0.00	-0.02	-0.03	0.01
who	-0.15	0.07	-0.17	0.37	-0.28	-0.16	0.51
at	-0.01	0.01	0.00	-0.00	-0.01	-0.00	-0.00
the	0.00	-0.03	-0.07	0.08	-0.02	-0.02	0.11
was	-0.01	-0.02	-0.02	0.04	0.00	-0.01	0.05

Table 4 shows the heat map of a one-channel model using the TREC-QA dataset (TREC-QA_5500_1ch model described in table 3). Cells represent contributions of words labeling table rows to classes labeling table columns. The intensity of a cell indicates the degree of the contribution of the corresponding word to the corresponding class. The dark red color indicates a strong negative contribution while the dark green color indicates a strong positive contribution to the class and the white color indicates almost null contribution. The last column (REL-HUM) represents the relevance to the predicted class (**HUM**) as the
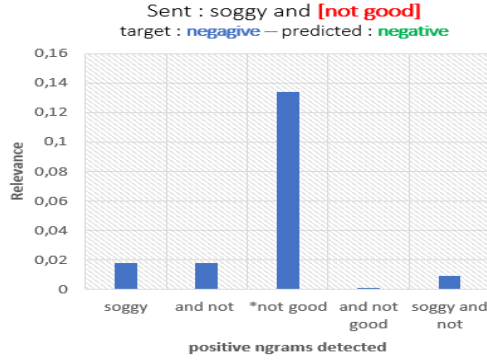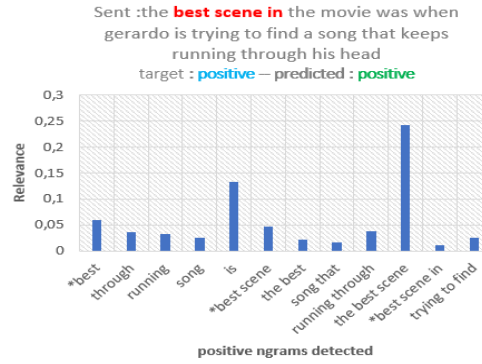
(a) A sample sentence from sent140
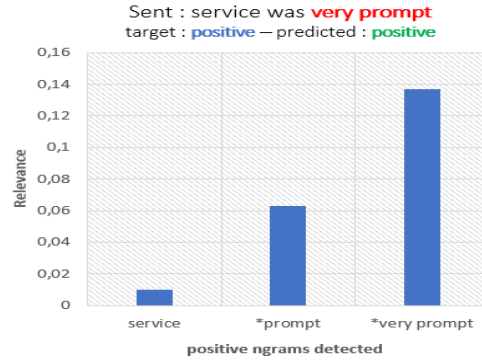
(b) A sample sentence from sent140
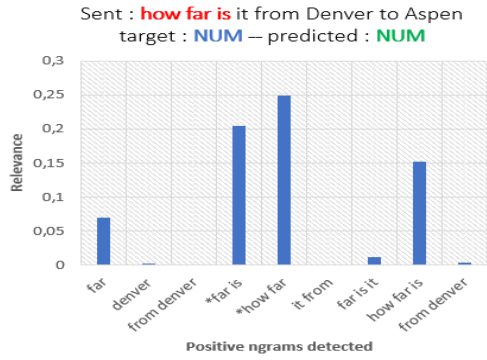
(c) A sample sentence from TREC-QA_5500

(d) A sample sentence from IMDB

Figure 3: Relevance of positive n-gram detected by filters.

difference between the contribution of a word to the class **HUM** and the mean of its contribution to the other classes as described in Sect. 5 Eq. 11. From this table, we can figure out that the term *"who"* does not only contribute highly to the class **HUM** but also contributes negatively to other classes except to **ENTY** where it contributes positively. By observing the relevance of each word to the predicted class, we can deduce that *"who was the"* are the main terms that were responsible for the prediction of the class **HUM**, which means that the expected type of the answer to that question is **HUMAN**.

6.3. *Discussion on sufficient and necessary set*

Figure 4 shows a list of sentences taken from a question answering dataset (**TREC-QA**) and from sentiment analysis datasets (**imdb**, **sent140**). In each sentence the value after the dashes "−" represents the class predicted by the model.

Question-Answering :

1) [When did] hawaii become a state ? – NUM

2) How tall is the building ? – NUM

3) Why does the moon turn orange ? – DESC

Sentiment Analysis :

4) It was either too cold not enough flavor or just bad – NEGATIVE

5) Not frightening in the least and barely comprehensible – NEGATIVE

6) Lately they have been extremely [nice] and helpful on the phone – POSITIVE

7) [Not] frightening in the least and very realistic – NEGATIVE

8) [Not] frightening in the least and [very realistic] – NEGATIVE

9) Not bad and very comprehensible – NEGATIVE

10) frightening in the least and [very realistic] – POSITIVE

Figure 4: Sufficient and necessary features

When this value is colored in green, it means that the model has correctly predicted the class, else the prediction is incorrect. The words colored in red in each sentence represent the sufficient n-gram features and those inside brackets represent necessary n-gram features. In sentence 1, the 2-gram *"When did"* was at the same time sufficient and necessary to predict the class **NUMBER** as the expected type of the answer to that question. In sentence 2, *"how tall"* was considered sufficient to predict the class **NUMBER** and there is no necessary feature.

In sentence 5, the term *"not"* was considered by the model as sufficient to predict the sentiment **NEGATIVE**. We performed little variations on that latter sentence, to change the underlying sentiment to POSITIVE, which have led to sentences 7, 8, 9, 10. We observe that the model has still predicted the sentiment NEGATIVE while still relying on the term *"not"*. However, when the term *"not"* is removed (sentence 10) the sentiment changes to POSITIVE, and the n-gram "very realistic" is determined as both sufficient and necessary to explain this new output. These examples show that in the case of sentences 7, 8, 9 the model has wrongly relied on the term *"not"* to produce its decision. This is most

likely due to insufficient training samples. Indeed, the model was built with a training set of 2250 sentences. A solution track would be to add in the training set, sentences which contain the term *"not"* with similar contexts to those of sentences where the prediction was marked as incorrect.

6.4. Comparison with LIME

Since LIME is a statistical model agnostic method for explanation which also evaluates the relevance of each input feature to the output classes, we thought of comparing the distribution of relevance using our method with that obtained using LIME. Figure 5 shows the distributions of relevance using LIME

(a) LIME and LRP-A on Sentiment Analysis   (b) LIME and LRP-A on TREC-QA dataset
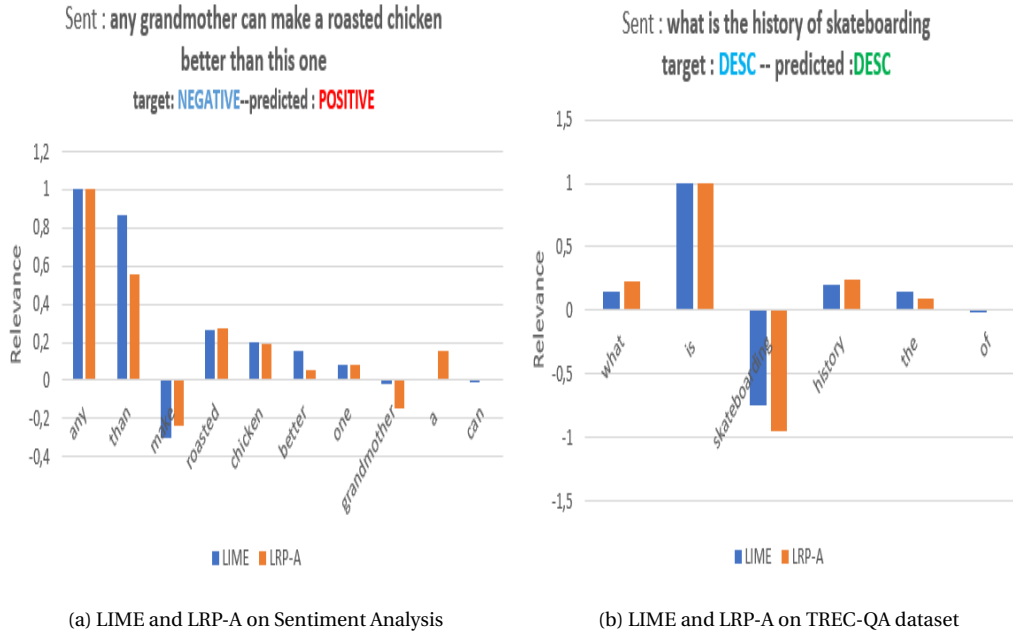
Figure 5: LIME versus CLRP

(in blue) and using our method (LRP-A) for two sentences. The first sentence (Figure 5a) is from **imdb** corpus and the second one (Figure 5b) is from the **TREC-QA** corpus. Datas have been normalized using Max-normalization in order to have both contributions in a comparable scale and to preserve the sign of each contribution. We observe that LIME and LRP-A (our method) have almost the same distribution of relevance among words for both sentences. The most important terms highlighted by both methods are the same.

However, we can observe that no filter has selected the term *"can"* with our method but LIME has still scored it though with low relevance. Since our method is a white box method, it is easy to verify that the term *"can"* had no impact on the output since it was not selected by any max-pooling filter. Another interesting fact to notice is that in Figure 5a, the model failed to predict the right class. Indeed, the model predicted the class **POSITIVE** while the expected class was the **NEGATIVE** class. When observing the distribution of relevance of words to the predicted class (**POSITIVE**) we can figure out that the model mainly relied on the features *(any, than)* which cannot be perceived as a correct justification for that prediction. Therefore, we can deduce that the model relied on wrong features to predict the class POSITIVE. This is likely due to an insufficient number of samples when training the model. This knowledge could provide insights to improve the model accuracy. For example, by providing negative samples where the terms *(any, than)* appear to tell the model not to associate them with the class **POSITIVE**.

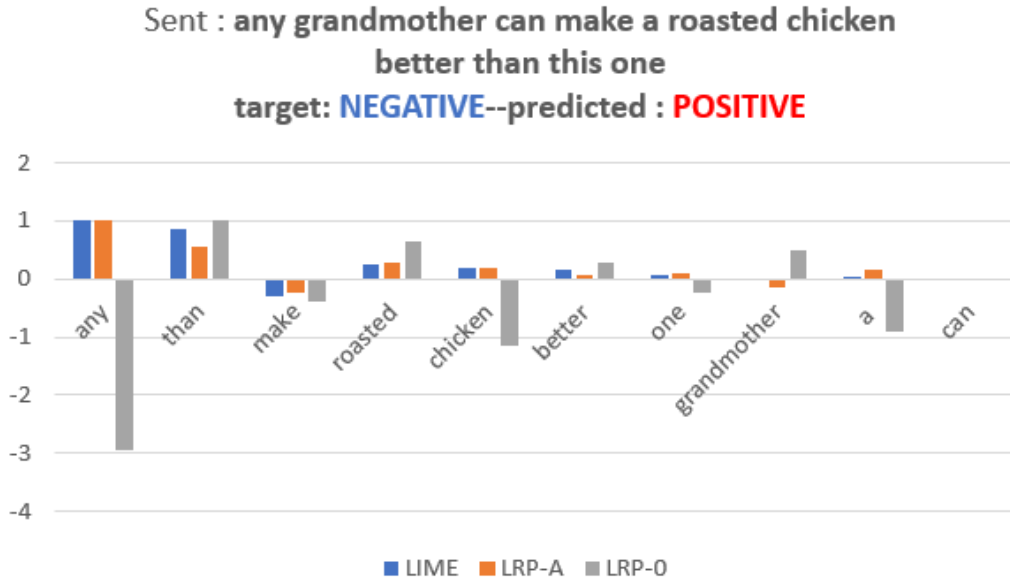### 6.5. *Impact of the LRP adaptation*

Figure 6: Impact of the new contribution ratio formula

As described in Section 5.1, to prevent undesired effects when the denominator in the LRP ratio (Eq. 4) is negative, we proposed to use Eq. 8 to compute

the ratio. Figure 6 presents a comparison between the distribution obtained using LIME, the standard LRP-0 and LRP-A. We observe that while LIME and LRP-A have almost the same distribution, the sign of the contribution of certain words when using LRP-0 diverge. This confirms the analysis done so far in section 5.1

7. Conclusion

We have described in this work, a method to explain 1D-Convolutional neural networks built for text classification. Experiments carried out on various datasets have shown that the distribution of relevance among the features matches with that obtained with a well-known statistical model agnostic method. In addition, the identification of sufficient and necessary features were shown to greatly simplify the explanation by highlighting key features. Techniques developed in this work can also be easily adapted to work with image processing using CNN.

References

[1] A. Adadi, M. Berrada, Peeking inside the black-box: A survey on explainable artificial intelligence (xai), IEEE Access 6 (2018) 52138–52160.

[2] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, D. Pedreschi, A survey of methods for explaining black box models, ACM Comput. Surv. 51 (2018) 93:1–93:42. URL: http://doi.acm.org/10.1145/3236009. doi:10.1145/3236009.

[3] S. Lowry, G. Macpherson, A blot on the profession, British medical journal (Clinical research ed.) 296 (1988) 657.

[4] T. Miller, Explanation in artificial intelligence: Insights from the social sciences, Artificial Intelligence 267 (2019) 1 – 38. URL: http://www.sciencedirect.com/science/article/pii/S0004370218305988. doi:https://doi.org/10.1016/j.artint.2018.07.007.

[5] A. B. Arrieta, N. D. Rodríguez, J. D. Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, F. Herrera, Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI, CoRR abs/1910.10045 (2019). URL: http://arxiv.org/abs/1910.10045. arXiv:1910.10045.

[6] A. Jacovi, O. Sar Shalom, Y. Goldberg, Understanding convolutional neural networks for text classification, in: Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, Association for Computational Linguistics, 2018, pp. 56–65. URL: `http://aclweb.org/anthology/W18-5408`.

[7] M. T. Ribeiro, S. Singh, C. Guestrin, "why should i trust you?": Explaining the predictions of any classifier, in: Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, ACM, New York, NY, USA, 2016, pp. 1135–1144. URL: `http://doi.acm.org/10.1145/2939672.2939778`. doi:10.1145/2939672.2939778.

[8] Y. Zhang, B. Wallace, A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification, arXiv preprint arXiv:1510.03820 (2015).

[9] A. Karpathy, J. Johnson, L. Fei-Fei, Visualizing and understanding recurrent networks, arXiv preprint arXiv:1506.02078 (2015).

[10] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, W. Samek, On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation, PLOS ONE 10 (2015) 1–46. URL: `https://doi.org/10.1371/journal.pone.0130140`. doi:10.1371/journal.pone.0130140.

[11] B. Carter, J. Mueller, S. Jain, D. Gifford, What made you do this? understanding black-box decisions with sufficient input subsets, in: The 22nd International Conference on Artificial Intelligence and Statistics, PMLR, 2019, pp. 567–576.

[12] E. Tjoa, C. Guan, A survey on explainable artificial intelligence (xai): Towards medical xai, ArXiv abs/1907.07374 (2019).

[13] D. Zeiler Matthew, F. Rob, Visualizing and understanding convolutional networks, CoRR.—-2013.—-Vol. abs/1311.2901.—-URL: http://arxiv.org/abs/1311.2901 (2013).

[14] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas, et al., Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav), in: International conference on machine learning, PMLR, 2018, pp. 2668–2677.

[15] N. Tishby, N. Zaslavsky, Deep learning and the information bottleneck principle, in: 2015 IEEE Information Theory Workshop (ITW), IEEE, 2015, pp. 1–5.

[16] R. Shwartz-Ziv, N. Tishby, Opening the black box of deep neural networks via information, arXiv preprint arXiv:1703.00810 (2017).

[17] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, Grad-cam: Visual explanations from deep networks via gradient-based localization, in: 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 618–626. doi:10.1109/ICCV.2017.74.

[18] P. Liu, A. A. Melkman, T. Akutsu, Extracting boolean and probabilistic rules from trained neural networks, Neural Networks 126 (2020) 300–311. URL: https://doi.org/10.1016/j.neunet.2020.03.024. doi:10.1016/j.neunet.2020.03.024.

[19] A. Ghorbani, J. Wexler, J. Y. Zou, B. Kim, Towards automatic concept-based explanations, in: Advances in Neural Information Processing Systems, 2019, pp. 9277–9286.

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

[21] L. M. Zintgraf, T. S. Cohen, T. Adel, M. Welling, Visualizing deep neural network decisions: Prediction difference analysis, arXiv preprint arXiv:1702.04595 (2017).

[22] P.-J. Kindermans, K. T. Schütt, M. Alber, K.-R. Müller, D. Erhan, B. Kim, S. Dähne, Learning how to explain neural networks: Patternnet and patternattribution, arXiv preprint arXiv:1705.05598 (2017).

[23] P. Angelov, E. Soares, Towards explainable deep neural networks (xdnn), Neural Networks 130 (2020). doi:10.1016/j.neunet.2020.07.010.

[24] D. Erhan, Y. Bengio, A. Courville, P. Vincent, Visualizing Higher-Layer Features of a Deep Network, Technical Report, Univeristy of Montreal, 2009.

[25] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1–9.

[26] P.-J. Kindermans, S. Hooker, J. Adebayo, M. Alber, K. T. Schütt, S. Dähne, D. Erhan, B. Kim, The (un) reliability of saliency methods, in: Explainable AI: Interpreting, Explaining and Visualizing Deep Learning, Springer, 2019, pp. 267–280.

[27] R. Caruana, Y. Lou, J. Gehrke, P. Koch, M. Sturm, N. Elhadad, Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission, in: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, 2015, pp. 1721–1730.

[28] B. Letham, C. Rudin, T. H. McCormick, D. Madigan, et al., Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model, The Annals of Applied Statistics 9 (2015) 1350–1371.

[29] S. Lapuschkin, S. Wäldchen, A. Binder, G. Montavon, W. Samek, K.-R. Müller, Unmasking clever hans predictors and assessing what machines really learn, Nature communications 10 (2019) 1–8.

[30] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, K.-R. Müller, Evaluating the visualization of what a deep neural network has learned, IEEE transactions on neural networks and learning systems 28 (2016) 2660–2673.

[31] L. Arras, F. Horn, G. Montavon, K.-R. Müller, W. Samek, " what is relevant in a text document?": An interpretable machine learning approach, PloS one 12 (2017) e0181142.

[32] Y. Kim, Convolutional neural networks for sentence classification, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Doha, Qatar, 2014, pp. 1746–1751. URL: `https://www.aclweb.org/anthology/D14-1181`. doi:10.3115/v1/D14-1181.

[33] K. F. F. Jiechieu, N. Tsopze, Skills prediction based on multi-label resume classification using cnn with model predictions explanation, Neural Computing and Applications (2020) 1–19.

[34] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, C. Potts, Learning word vectors for sentiment analysis, in: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, Portland, Oregon, USA, 2011, pp. 142–150. URL: `http://www.aclweb.org/anthology/P11-1015`.

[35] A. Go, R. Bhayani, L. Huang, Twitter sentiment classification using distant supervision, CS224N project report, Stanford 1 (2009) 2009.

[36] E. Voorhees, The trec question answering track, Nat. Lang. Eng. 7 (2001) 361–378. doi:`10.1017/S1351324901002789`.

[37] D. Kotzias, M. Denil, N. De Freitas, P. Smyth, From group to individual labels using deep features, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015, pp. 597–606.