Report: *"What is Relevant in a Text Document?": An Interpretable Machine Learning Approach*

Author: Christoph Schaller

Nr. 3369103
Course: Explainable Machinelearning

---

1. Motivation

Rising capabilities of machine learning algorithms and neural networks make it possible to analyse, annotate and process data in a magnitude far above human ability. But can we trust these algorithms? How is it possible to make sure, that the decisions to categorize and classify documents are based on valid features and not on features only present in our training and testing sets? As a small oversight in preprocessing or acquisition of data can lead to failure in a real classification task by promoting features that are overrepresented in the training and test data but usually not found *in the wild*. Thereby it is good to know on which features the network or algorithm bases its decision. A way to extract this information from a trained model is layer-wise relevance propagation. With this technique it is possible to visualize and explain the predictions of complex non-linear classifiers. The Paper *"What is Relevant in a Text Document?": An Interpretable Machine Learning Approach* [1] of Leila Arras et al. brings this technique from image recognition to text categorisation. And tries to find measurements to tell how good a machine learning model explains a text category.

2. Related Work

There are different approaches to explain the decisions a machine learning model makes and what kind of features it depends these decisions on. While some approaches aim to explain the inner workings of single algorithms or networks, others try to explain different models by using random sampling but impose an additional computational cost[2][3][4]. The method chosen by Arras et al. called LRP short for layer-wise relevance propagation is an approach applicable to different models including deep neural

networks and support vector machines. LRP was beforehand used to display heatmaps to highlight important areas in images for computer vision.[1]

3. Predicting the Class of a Document with different Approaches

3.1. Embeddings

An important difference between the CNN and SVM are their respective word embeddings. While SVM is using a bag of words model, the CNN is, in this case, using a continuous bag of words model. The BoW (bag of words) model is representing the corpus of documents its trained on by a simple set of (usually) all context words contained in it, single documents are then represented as a vector where each word is represented by how often it occured in the document. Compared to the BoW model the CBoW model is a bit more complicated, here for a single word that should be represented as a feature all context words in its vicinity, so in a window of a fixed size are used. These context words are then used as features to train a shallow single layer neural network to predict a word by the context word often found in its surroundings. The vectors for the respective words are then taken from the last layer of the shallow network and used as embeddings. This allows the embeddings to model the meaning of certain words, as words with a related meaning are often found in the neighborhood of the same group of other words. This works so well that it is possible to convey certain meanings in a vector, taking a vector between the embeddings of the words *"man"* and *"king"* and applying it to the embedding for the word *"woman"* will result in a location near the embedding for *"queen"*. So while the CNN model can take advantage of the encoded semantic similarity of the CBoW embeddings the SVM models BoW embeddings are equidistant in its semantic space.[1]

3.2. CNN

To train the CNN model for text document categorization first every word in a document gets mapped to its word2vec (CBoW) vector. It is important to note that the CBoW embeddings were not trained on the document corpus but on a larger dataset containing far more different words and a larger amount of text. The document representation a matrix of the number of words times the size of the embedding vectors, is then fed into the convolutional neural network which convolutional layers are filtering down the input data. Subsequently the next layer of the network computes a maximum for each representation over all dimensions of the document. At the end all the now pooled features are fed into a logistic classifier returning

the unnormalized log-probabilities for all possible document classes. After converting these outputs with a softmax function to probabilities, the classification is done.[1]

3.3. SVM

For the training of the SVM classifier, the bag of words embeddings are used, for this each document is represented by a vector as long as the size of the training data vocabulary. This means that every word, in all documents, is in this set. For a single document each word is mapped as its term-frequency-inverse-document-frequency score. This is the frequency of the given word in the current document, compared to the inverse of the words frequency in the whole corpus. TF-IDF increases the weight of words which are meaningful in the context of the whole corpus and decreases the weight of words that are very common throughout all of the documents. The vectors of TF-IDF scores are then normalized to euclidean norm and using all the document vectors, hyperplanes separating different document classes are learned. This results in a linear prediction score for each class of documents.[1]

4. Decomposing the Learning

The importance of single features for the classification of a document can be calculated by using layer-wise relevance propagation. As this method can be used to deconstruct convolutional neural networks as well as support vector machines, it is possible to directly compare the results for both approaches. This works by redistributing the score that caused a classification onto the respective input neurons via backward propagation until the input space is reached and the scores can be mapped to input features.[1]

4.1. Decomposing the CNN with LRP

To decompose the inner workings of the CNN based classifier the unnormalized classification scores are used to start, then for each layer of the network, according to their relevance in the classification a score is assigned to each of the layers neurons. So for each CNN neuron $x_{i,t}, x_{j,t}, x_j, x_k$ there is a relevance score $R_{i,t}, R_{j,t}, R_j, R_k$, it is important to note that the relevance per class mapped to the neurons on each layer is the size of the score the classifier predicted for the class, this can be expressed as:

$$\sum_{i,t} R_{i,t} = \sum_{j,t} R_{j,t} = \sum_{j} R_j = \sum_{k} R_k \tag{1}$$

The relevance redistribution is then formalized by introducing an new concept, how much relevance is propagated from a neuron b to a neuron a in the next lower level is indicated by messages. The top layer relevance vector is then set as:

$$\forall_k : R_k = X_k \cdot \delta_{kc} \tag{2}$$

While the messages in the top fully-connected layer are distributed by the following weighted formula, where z_{jk} represents the share of each neurons contribution to the upper neurons weight in the forward propagation. A small stabilizing term $\epsilon = 0.01$ is then added to prevent the denominator from nearing zero and avoid too large or small relevance messages.

$$R_{j \leftarrow k} = \frac{Z_{jk}}{\sum_j Z_{jk}} R_k \tag{3}$$

The calculated distribution of messages is then pooled onto the respective neurons with:

$$R_j = \sum_k R_{j \leftarrow k} \tag{4}$$

And the relevance scores R_j are propagated through the max-pooling layer with a *"winner-takes-all"* redistribution to acknowledge the rule used for backpropagation while training.

$$R_{j,t} = \begin{cases} R_j & \text{if } t = \text{argmax}_{t'} X_{j,t'} \\ 0 & \text{else} \end{cases} \tag{5}$$

For the final convolutional layer the weighted redistriubtion is done with:

$$R_{(i,t \leftarrow \tau) \leftarrow (j,t)} = \frac{Z_{i,j,\tau}}{\sum_{i,\tau} Z_{i,j,\tau}} \tag{6}$$

Which is similar to the distribution of messages in a fully connected layer except for the added notational complexity to make up for the convolutional nature of the layer. The messages are then pooled onto the input neurons with:

$$R_{i,t} = \sum_{j,\tau} R_{(i,t)\Leftarrow(j,t+\tau)} \tag{7}$$

[1]

4.2. Word Relevance for the CNN

As the relevance for predicting a certain class is distributed onto the input features, it still needs to be pooled onto the words represented by the embeddings to obtain a per word relevance score.

$$R_t = \sum_i R_{i,t} \tag{8}$$

is pooling the relevances onto all dimensions of the CBoW vector. Besides that it is possible to use these word relevance scores to condense the semantic information of a text document to a single vector in the same vector space as the CBoW vectors. This is done by linearly combining the vectors of the word in the document with what is called word-level extraction by Arras et al.:

$$\forall_i : d_i = \sum_t R_t \cdot x_{i,t} \tag{9}$$

To avoid this step of word-level pooling it is possible to extract only the relevant subspace for each word, adressing the problem of word homonymy and resulting in a finer grained semantic representation of the summarized document by what is called element-wise extraction by Arras et al.:

$$\forall_i : d_i = \sum_t R_{i,t} \cdot x_{i,t} \tag{10}$$

[1]

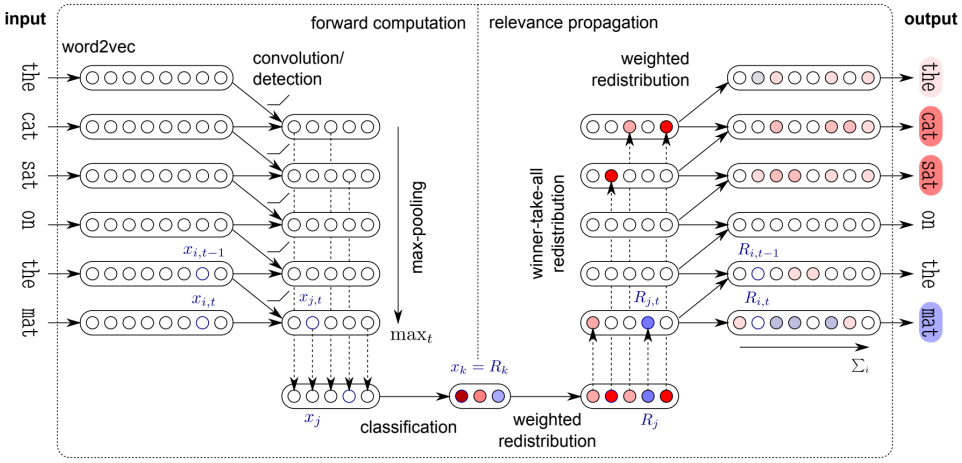

Figure 1: Diagram of a CNN-based interpretable machine learning system consisting of a forward processing that computes for each input document a high-level concept (e.g. semantic category or sentiment), and a redistribution procedure that explains the prediction in terms of words. Arras et al. [1]

4.3. Decomposing the SVM with LRP

The decomposition of the support vector machine is compared to the decomposition of the CNN much simpler. As the SVM with BoW embeddings is only a linear predictor the relevance distribution for a class c can be computed with the following formula.

$$R_{i,t} = \sum_{j,\tau} R_{(i,t)\Leftarrow(j,t+\tau)} \tag{11}$$

Where D is the number of non zero entries of the vector x.[1]

4.4. Document Summary for the SVM

To summarize a document with the results of the SVM classifier it is possible to use the following formula:

$$\forall_i : d_i = \sum_t R_i \cdot \tilde{x_i} \tag{12}$$

For further accomodation of the two document summarization methods for the CNN it is also possible to swap R_t in the formula for the word-level extraction with the inverse document frequency score of the document and swap R_i in the element-wise extraction formula for a TF-IDF score.[1]

4.5. Sensitivity Analysis as a Way of Decomposing ML-Models

As a Baseline to the layer-wise relevance propagation Arras et al. use sensitivity analysis. Each input feature is assigned a score representing the steepness of the decision function in the input space. This local steepness is a weak representation of the actual function value that is the main interest when decomposing the inner workings of a machine learning model.[1]

5. Measuring the Quality of Word Relevances

The methods to obtain a relevance distribution over words are evaluated by intrinsic validation, while the performance of the machine learning algorithms decompositions to explain a model is analyzed by extrinsic validation.[1]

5.1. Intrinsic Validation

The quality of the word relevance scores is measured by creating and analyzing heatmaps of words labeled important for a class in a document by the decomposition algorithms. Another way to evaluate the quality is composing a list of the most important words for a class over all documents, as this allows to inspect which words are taken as the most important ones for the general classification. To quantify the relevance of a word across different classifiers and decomposition algorithms Arras et al. remove words from the document and measure the impact on the classification scores. The idea behind this is, that words more important to the classification process will impact the scores more than words that are not as important to the classification. By removing words in different orders, for example declining from the word tagged as most relevant or ascending from the word tagged most irrelevant it is possible to analyze how precise the decomposing algorithm is in classifying the importance of words for a single machine learning approach.[1]

5.2. Extrinsic Validation

Extrinsic validation is used to compare the explanatory power of different machine learning approaches, this is not possible with the method of intrinsic validation, as the removal of words from documents affects different machine learning algorithms and neural networks in different ways. Thereby Arras et al. define a model as more explainable if its word relevances are more semantic extractive. They define this as helpful to solving semantic related tasks such as the classification of document summary vectors. Arras et al. define three steps to analyze a decomposed model this way.

- First they generate document summary vectors for all documents in the test set, the predicted class is used as the target class for the decomposition.
- Second the document summary vectors are normalized to euclidean norm and a K-nearest-neighbors classification of half the vectors, with the other half as neighbors, is done. The neighbor votes are weighted uniformly and the classification is done for different K.
- Finally the second step is repeated for ten random data splits and the mean KNN classification accuracy for the best hyperparameter K is set as explanatory power index or EPI. If the explanatory power index is high, the better the machine learning model and the respective decomposition algorithm are, in explaining the model.

Arras et al. claim that using the KNN as an external classifier allows them to objectively compare different machine learning models. To analyze their density and the local neighborhood structure of the semantic information, that was extracted with the document summary vectors.[1]

6. Results and Evaluation

6.1. Document Classification

To analyze the performance of the different machine learning approaches to classify documents, Arras et al. compare three CNNs with different filter sizes and the support vector machine with bag of words embeddings. The dataset Arras et al. use for training and testing is the 20Newsgroups dataset, which consists of newsgroup posts that are evenly distributet between 20 different classes. In the preprocessing, headers are removed from the documents and the text is tokenized with NLTK (natural language toolkit). After tokenization, only words containing alphabetic characters, hyphens, dots and apostrophes and words containing at least one alphabetic character are kept, this removes punctuation, numbers and dates. Additionally, but only for the SVM classifier all words are converted to lowercase to allow compatibility with the bag of words embeddings. Finally the first 400 tokens in a document are used, this is done to simplify training even if the models could theoretically work with documents of any size. While the SVM uses a BoW of the words of all documents the CNNs CBoW embeddings originate from 300-dimensional freely available word2vec embeddings, words that are not in the vocabulary are simply initialized as zeros. The SVM is tuned with tenfold-crossvalidation but Arras et al. perform

no grid search to fine tune the hyperparameters of the CNN classifier and stopped when obtaining reasonable results. [1]

Table 1: Performance of the different classifiers.

ML Model	Test Accuracy (%)
Bow/SVM (V = 70631 words)	80.10
CNN1 (H = 1, F = 600)	79.79
CNN2 (H = 2, F = 800)	**80.19**
CNN3 (H = 3, F = 600)	79.75

6.2. Identifying Relevant Words

Arras et al. compile the results of the decomposition of the SVM and CNN models into heatmaps, the documents chosen for this are part of the classes sci.space and sci.med. Analyzing the heat maps the words considered important by the CNN appear to be much sparser than the words considered relevant by the SVM, additionally the words considered important by the SVM are mostly insignificant words like the, is, of that appear in great number in every document while the most important words for the CNN are mainly semantically meaningful words. This can be explained by taking into account that for the SVM relevance values are computed for the whole bag of words feature, relying completely on local and global word statistics, while the CNN can use the words position, neighborhood and the knowledge encoded in CBoW embeddings to its advantage.

To get the words deemed most important for the classification of a certain class Arras et al. set a class as the target class for decomposition of a model and decompose for all documents in the test set. From the resulting words the most relevant thirty are selected to represent the class. Again it is visible that SVM assignes relevance to insignificant words. The underlined words in the figures are not present in the training set but are related to similar words in the training corpus and thereby get relevance distributed onto them by their closeness in the CBoW vector space. As this is not possible with the BoW embeddings of the SVM there are no underlined words in the tables for the SVM.[1]

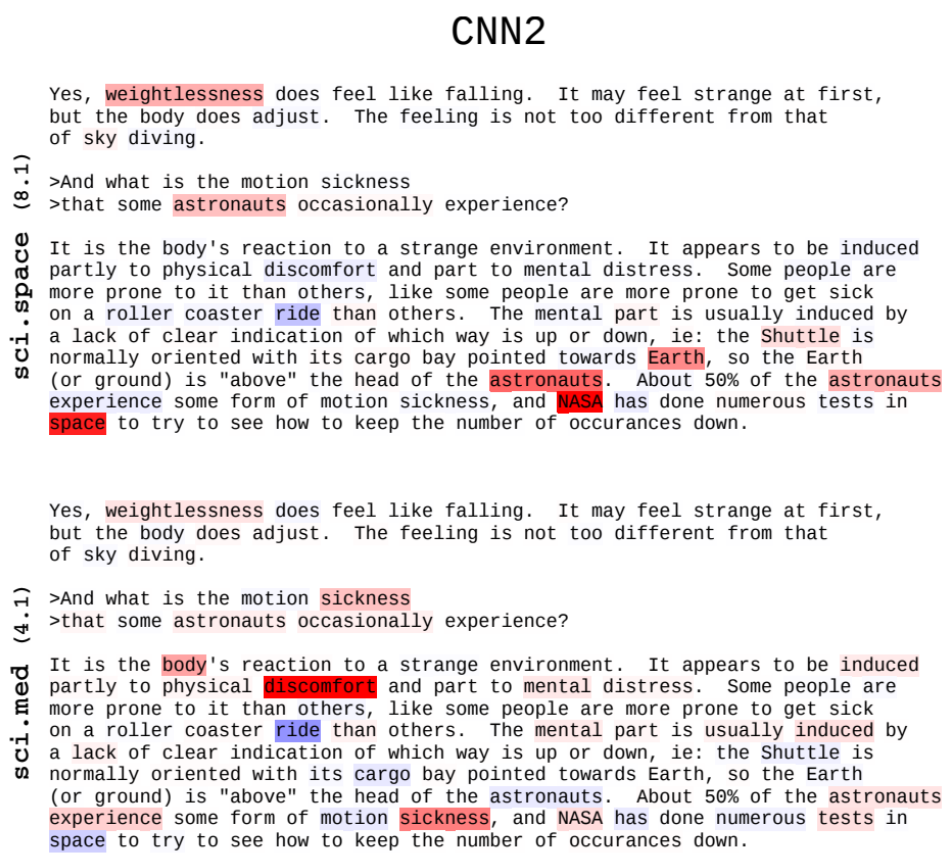

Figure 2: LRP heatmaps of the document sci.space 61393 for the CNN2 model. Positive relevance is mapped to red, negative to blue. Arras et al. [1]

6.3. Document Summary Vector

Comparing the different document summary vectors it is possible to clearly differentiate between summary vectors created by different methods. While the documents summarized with uniform or TFIDF are clumped together and clusters are barely discernible. The clusters formed by the summary with sensitivity analysis decomposition are clearly separated and the clusters formed by documents summarized with the layer-wise relevance

Figure 3: LRP heatmaps of the document sci.space 61393 for the SVM model. Positive relevance is mapped to red, negative to blue. Arras et al. [1]

propagation decomposition are the most clearly separated ones. This suggests, again that LRP provides more powerful semantically extraction than sensitivity analysis, but SA is still superior to TFIDF and uniform semantic extraction.[1]

6.4. Quantitative Evaluation

Analyzing the results of the extrinsic validation shows the impact that the deletion of words deemed important or unimportant have to the classification. Deleting words deemed important in correctly classified documents, causes LRP to lose accuracy first rapidly and then in a slower pace, while SAs accuracy loss almost comes to a halt after a significantly shorter initial drop.

sci.med	sci.space	comp.graphics
symptoms (7.3), treatments (6.6), medication (6.4), osteopathy (6.3), ulcers (6.2), sciatica (6.0), hypertension (6.0), herb (5.6), doctor (5.4), physician (5.1), Therapy (5.1), antibiotics (5.1), Asthma (5.0), renal (5.0), medicines (4.9), caffeine (4.9), infection (4.9), gastrointestinal (4.8), therapy (4.8), homeopathic (4.7), medicine (4.7), allergic (4.7), dosages (4.7), esophagitis (4.7), inflammation (4.6), arrhythmias (4.6), cancer (4.6), disease (4.6), migraine (4.6), patients (4.5).	spacecraft (11.0), orbit (10.8), NASA (8.6), Mars (7.8), moon (7.1), orbiting (7.1), Martian (6.8), orbital (6.8), shuttle (6.7), SMOS (6.6), telescope (6.5), Space (6.5), rocket (6.3), GRBs (6.0), Earth (6.0), astronaut (5.9), Moon (5.7), Shuttle (5.7), lander (5.6), Flyby (5.3), planets (5.2), Hubble (5.2), Soyuz (5.2), geosynchronous (5.2), Endeavour (5.1), space (5.0), planetary (4.9), Nasa (4.9), Astronomy (4.9), astronauts (4.9).	Graphics (6.9), raytracing (6.8), graphics (6.8), polygon (6.5), animation (6.3), Image (6.2), shaders (6.2), pixel (5.7), fractal (5.5), viewports (5.5), Autodesk (5.4), visualization (5.2), RGB (5.1), images (5.0), TIFF (5.0), Corel (4.9), Studio (4.9), algorithm (4.8), Bezier (4.8), polygons (4.7), GIF (4.7), Pixel (4.6), algorithms (4.5), modo (4.5), image (4.4), radiosity (4.4), AutoDesk (4.3), Studios (4.3), HPGL (4.2), JPEG (4.2).

Figure 4: The 30 most relevant words per class for the CNN2 model listed in decreasing order of their relevance (value indicated in parentheses). Underlined words do not occur in the training data. Arras et al. [1]

sci.med	sci.space	comp.graphics
cancer (1.4), photography (1.0), doctor (1.0), msg (0.9), disease (0.9), medical (0.8), sleep (0.8), radiologist (0.7), eye (0.7), treatment (0.7), prozac (0.7), vitamin (0.7), epilepsy (0.7), health (0.6), yeast (0.6), skin (0.6), pain (0.5), liver (0.5), physician (0.5), she (0.5), needles (0.5), dn (0.5), circumcision (0.5), syndrome (0.5), migraine (0.5), antibiotic (0.5), water (0.5), blood (0.5), fat (0.4), weight (0.4).	space (1.6), launch (1.4), ics.uci.edu (1.2), moon (1.1), orbit (1.0), mars (1.0), pat (1.0), nasa (0.9), dietz (0.9), shuttle (0.8), solar (0.7), command (0.7), henry (0.6), fred (0.6), gamma (0.6), sci.space (0.6), pluto (0.6), satellite (0.6), dc-x (0.6), nicho (0.6), astronomy (0.5), lunar (0.5), pom (0.5), hga (0.5), sky (0.5), spacecraft (0.5), gravity (0.5), scicom.alphacdc.com (0.5), nick (0.4), roland (0.4).	graphics (2.0), phigs (1.4), image (1.4), images (1.4), xv (1.3), tiff (1.2), polygons (1.1), comp.graphics (1.0), mpeg (1.0), format (1.0), siggraph (1.0), povray (0.9), quicktime (0.8), bockamp (0.8), surface (0.8), animation (0.8), iges (0.8), studio (0.8), jpeg (0.8), pov (0.7), dec (0.7), scodal (0.7), algorithm (0.7), genoa (0.7), sgi (0.7), palette (0.6), vga (0.6), impulse (0.6), c (0.6), rgb (0.6).

Figure 5: The 30 most relevant words per class for the BoW/SVM model listed in decreasing order of their relevance (value indicated in parentheses). Underlined words do not occur in the training data. Arras et al. [1]

This indicates that LRP is able to identify relevant words better and more reliable.

A similar behavior is seen when deleting the least relevant words from wrongly classified documents, while LRP gains accuracy in a first steep and later declining pace, SA is gaining almost as little accuracy as the randomized baselines. All in all LRP outperforms SA and the randomized baselines in every task. Another interesting discovery with the extrinsic validation method is the difference between varying filter sizes of the CNNs, as networks with larger filter sizes are more sensitive to word deletions. Arras et al. guess that this is the case because the meaning of surrounding words becomes less obvious to classifiers when words are deleted in their neighborhood.[1]

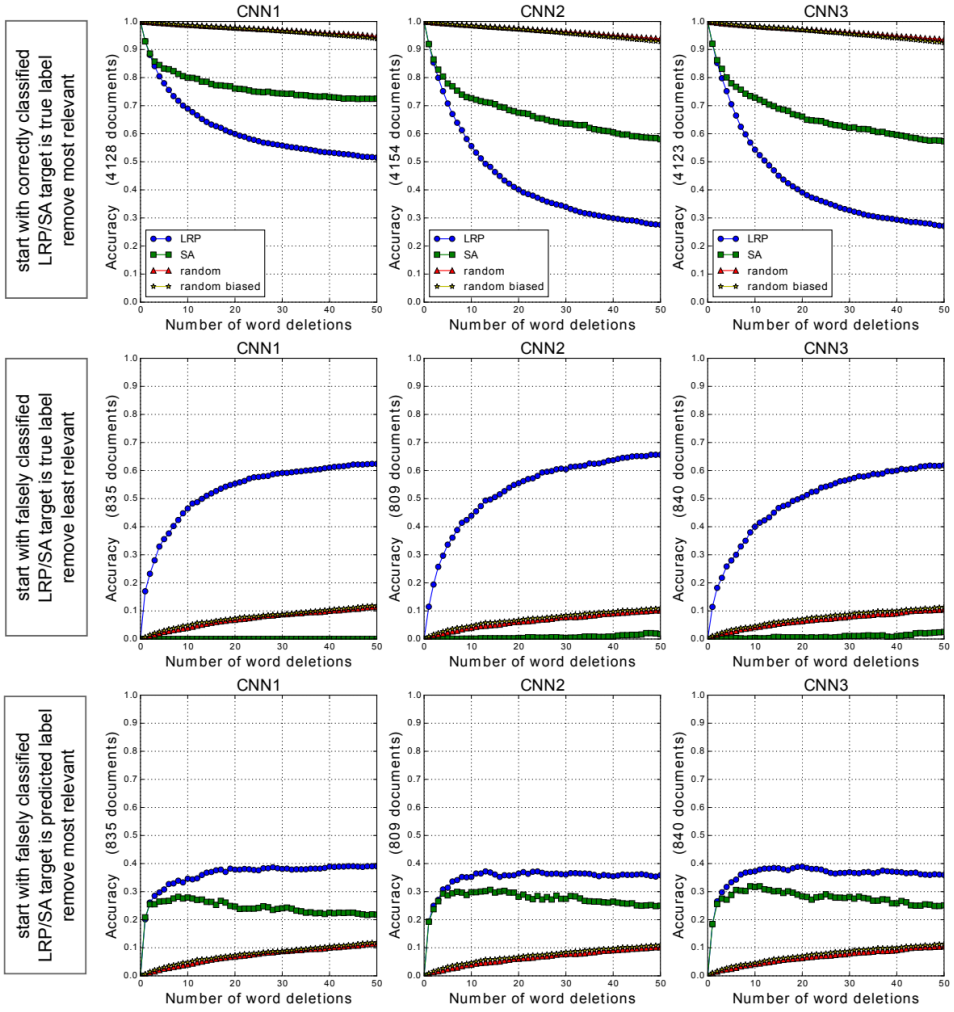

Figure 6: KNN accuracy when classifying the document summary vectors of half of the 20Newsgroups test documents (other half is used as neighbors). Results are averaged over 10 random data splits. Arras et al. [1]

6.5. Explanatory Power

Applying the method proposed to measure explanatory power, Arras et al. are abled to show that the LRP decompositions with element-wise summarys of CBoW vectors significantly outperform the other approaches when comparing the explanatory power index. While the performance of LRP is still better than the respective non element-wise SA and TFIDF document representation.[1]

Semantic Extraction		Explanatory Power Index (EPI)	KNN parameter
word2vec/CNN1	LRP (ew)	0.8045 (0.0044)	K = 10
	SA (ew)	0.7924 (0.0052)	K = 9
	LRP	0.7792 (0.0047)	K = 8
	SA	0.7773 (0.0041)	K = 6
word2vec/CNN2	LRP (ew)	0.8076 (0.0041)	K = 10
	SA (ew)	0.7993 (0.0045)	K = 9
	LRP	0.7847 (0.0043)	K = 8
	SA	0.7767 (0.0053)	K = 8
word2vec/CNN3	LRP (ew)	0.8034 (0.0039)	K = 13
	SA (ew)	0.7931 (0.0048)	K = 10
	LRP	0.7793 (0.0037)	K = 7
	SA	0.7739 (0.0054)	K = 6
word2vec	TFIDF	0.6816 (0.0044)	K = 1
	uniform	0.6208 (0.0052)	K = 1
BoW/SVM	LRP	0.7978 (0.0048)	K = 14
	SA	0.7837 (0.0047)	K = 17
BoW	TFIDF	0.7592 (0.0039)	K = 1
	uniform	0.6669 (0.0061)	K = 1

Table 2: Results averaged over 10 random data splits. For each semantic extraction method, we report the explanatory power index (EPI) corresponding to the maximum mean KNN accuracy obtained when varying the number of neighbors K, the corresponding standard deviation over the multiple data splits, and the hyperparameter K that led to the maximum accuracy. Arras et al. [1]

7. Conclusion

Arras et al. are able to show what layer-wise relevance propagation is capable to achieve when applied to machine learning models for text classification and propose with their methods of intrinsic and extrinsic validation powerful methods to analyze different classifiers and the methods to decompose them. Allowing a glance behind the facade of CNNs and SVMs, resulting in easier understanding and increased explainability of the analyzed machine learning models.[1]

[1] L. Arras, F. Horn, G. Montavon, K. Müller, W. Samek, "what is relevant in a text document?": An interpretable machine learning approach, CoRR abs/1612.07843 (2016).

[2] E. Strumbelj, I. Kononenko, An efficient explanation of individual classifications using game theory 11 (2010) 1–18.

[3] M. T. Ribeiro, S. Singh, C. Guestrin, "why should I trust you?": Explaining the predictions of any classifier, CoRR abs/1602.04938 (2016).

[4] R. Turner, A model explanation system: Latest updates and extensions (2016).