Evaluating the Performance of Interpretability Methods in Text Categorization Task

Alisher Rogov[1] and Natalia Loukachevitch[2]

[1] Bauman Moscow State Technical University, Moscow, Russia
rogov.alisher@gmail.com
[2] Lomonosov Moscow State University, Moscow, Russia
louk_nat@mail.ru

Abstract. Neural networks are progressively assuming a larger role in individuals daily routines, as their complexity continues to grow. While the model demonstrates satisfactory performance when evaluated on the test data, it often yields unforeseen outcomes in real-world scenarios. To diagnose the source of these errors, understanding the decision-making process employed by the model becomes crucial. In this paper, we consider various methods of interpreting the BERT model in classification tasks, and also consider methods for evaluating interpretation methods using vector representations fastText, GloVe and Sentence-BERT.

Keywords: Interpretability · BERT · Classification.

1 Introduction

As neural networks advanced, they increasingly achieved results comparable to human performance and even surpassed them in certain domains. However, this progress coincided with the growing complexity of models, characterized by an exponential increase in the number of parameters employed in the network. Deep learning models themselves are not easy to interpret because of their "black box" nature, and with such development it is almost impossible. The benefit of interpretability in machine learning is that it increases the credibility of the model. People are often afraid to rely on machine learning models when solving certain critical tasks. In particular, there may be situations when a person is faced with a new technology, and such an attitude can slow down the pace of its implementation.

Interpretability methods can be evaluated from three points of view: application-grounded, functionally-grounded or human-grounded [12, 3]. Application-grounded evaluation estimates consequences in the target environment, for example explanations in bank services. Functionally-grounded evaluation aims to check how well the explanation reflects the model. Human-grounded evaluation estimates if the explanations are understandable to humans.

In this paper, we consider the post-hoc interpretation methods in the text categorization task and suppose that to be human-grounded the explanation should be semantically related to the category's name. A user should see semantic

similarity between explanation and the category. We compare several known ways of interpreting the results of the deep learning model: LIME [15], SHAP [11] and the self-attention mechanism of BERT as a way of interpreting the results [6] using this approach.

The interpretation methods considered in this paper yield a ranked list of words with corresponding weights, which indicate the contribution of each word to the model's decision-making process. Subsequently, we select the top N words, where N can take values from the set $\{1, 3, 5, 10\}$, that have contributed most positively to the classification output of the interpreted model. In order to compare these interpretation results with the category name, we employ two approaches.

In first approach, we utilize word and sentence embeddings, along with the Normalized Discounted Cumulative Gain (NDCG) metric from information retrieval [7]. This allows us to evaluate the semantic similarity between the interpretation results and the category name.

In second approach, we pass the entire interpretation result as a sentence to the Sentence-BERT model, which generates a single embedding representation. We then compare these selected words as sentence with the category name in terms of their semantic similarity using embeddings.

2 Related Work

Various explanation methods have been proposed to address the need for interpretability of machine-learning methods. However, it is quite difficult to understand what a method is most trustworthy. Yalcin and Fan [16] analyzed explanations given by SHAP and LIME methods for classification of poisonous mushrooms in the mushroom dataset. They found that for more than a third of samples, SHAP and LIME give different explanations when comparing the most important feature.

The authors of [4] study local-explanation methods on a wide range (304) of OpenML datasets using six quantitative metrics. They revealed that LIME and SHAP's approximations are particularly efficient in high dimension and generate intelligible global explanations, but they suffer from a lack of precision regarding local explanations.

Natural language processing tasks have their own specificity, therefore the approaches to their interpretability should be studied separately.

In [9] the authors consider several intepretability methods in text categorization. Three tasks for evaluating intepretability were considered: 1) determining the best classification model from several ones based on explanations; 2) identifying the category of an example based on explanation; 3) help in analysis of examples with low probabilities. It was find that the LIME method obtained the best results in the second task, it finds the best evidence for the class independent of the class correctness. The study was implemented for two text categorization datasets (Amazon reviews and arxiv papers) and involved crowdsourcers in the first case and post-graduate students in the second case.

Our study provides automatic evaluation of the task 2 defined in the above-mentioned paper [9]. We calculate semantic similarity of words extracted by interpretability methods with the category's title.

3 Interpretation Algorithms

In our experiments we consider three known algorithms of interpretation: LIME [15], SHAP [11] and self-attention weights [6].

3.1 LIME

LIME [15] (Local Interpretable Model-agnostic Explanations) is a method of local interpretation independent of the machine learning model. Local interpretability implies knowing the reasons for a specific decision. LIME presents a locally faithful explanation by fitting a set of perturbed samples near the target sample using a potentially interpretable model, such as linear models and decision trees [10]. The interpreted explanation in LIME is presented in the form of a binary vector showing the participation of any parameter in the result. For example, a possible interpretable representation for text classification is a binary vector indicating the presence or absence of a word, even though the classifier may use more complex (and incomprehensible) features such as word embeddings [15].

Let $x \in R^d$ be the instance being explained, the explained model be denoted by $f : R^d \to R$ and the explanation of the model is presented as a model $g \in G$, where G is a class of interpreted models, such as linear models

$$g(x') = \phi_0 + \sum_{i=1}^{d'} \phi_i x'_i \tag{1}$$

where $x' \in \{0,1\}^{d'}$ is a binary vector of interpretable representation of x and $\phi_i \in R$.

Interpreted model g tries to ensure $g(z') \approx f(x)$ whenever $z' \approx x'$. Since not every $g \in G$ can be simple enough to be interpreted, a measure of the complexity $\Omega(g)$ of the explanation of $g \in G$ is introduced. For linear models, $\Omega(g)$ may be the number of non-zero weights. Next, $\pi_x(z)$ is used as a measure of proximity between the perturbed sample z and x. $L(f, g, \pi_x)$ will be a measure of how incorrectly g approaches f in the locality defined by π_x.

$$\xi(x) = \arg\min_{g \in G} L(f, g, \pi_x) + \Omega(g). \tag{2}$$

Thus, the essence of the LIME approach is that we approximate the prediction of the model f of the test case x by a simpler, easily interpreted model g, which uses a simplified representation. The resulting explanation $\xi(x)$ interprets the target sample x with linear weights when g is a linear model.

3.2 SHAP

SHAP[11] (SHapley Additive exPlanations) is a game theoretic approach to explain the output of any machine learning model. It connects optimal credit allocation with local explanations using the classic Shapley values from game theory and their related extensions. Shapley regression values are important characteristics for linear models in the presence of multicollinearity. To calculate the Shapley value, it is required to retrain the model for all subsets of features $S \subseteq F$, where F is the set of all features. These values assign an importance value to each feature, which means the importance of including this feature in the model forecast.

To calculate the Shapley value, the $f_{S\cup\{i\}}$ model is trained with the presence of this feature, and the other f_S model is trained without the feature. Then the predictions from the two models are compared at the current input $f_{S\cup\{i\}}(x_{S\cup\{i\}}) - f_S(x_S)$, where x_S represents the values of the input features in the set S. The feature retention depends on other features in the model, the previous differences are calculated for all possible subsets of $S \subseteq F \setminus \{i\}$. Shapley values are a weighted average of all possible differences:

$$\phi_i = \sum_{S\subseteq F\setminus\{i\}} \frac{|S|!(|F|-|S|-1)!}{|F|!}[f_{S\cup\{i\}}(x_{S\cup\{i\}}) - f_S(x_S)]. \tag{3}$$

The exact computation of Shapley values is challenging. The work [11] introduces a new perspective that unifies Shapley value estimation. They propose SHAP values, that are the Shapley values of a conditional expectation function of the original model. Let $x \in R^d$ be the instance being explained, and $x' \in \{0,1\}^M$ will denote a binary vector for its interpretable representation and h be the mapping function $x = h_x(x')$. SHAP values are the solution of

$$\phi_i(f,x) = \sum_{z'\subseteq x'} \frac{|z'|!(M-|z'|-1)!}{M!}[f(h_x(z')) - f(h_x(z'\setminus i))] \tag{4}$$

where $z' \in \{0,1\}^M$, $z' \setminus i$ denote setting $z'_i = 0$, $|z'|$ is the number of non-zero entries in z', and $z' \subseteq x'$ represents all z' vectors where the non-zero entries are a subset of the non-zero entries in x'. In [11] authors propose that $f(h_x(z')) = E[f(z)|z_S]$, where S is the set of non-zero indexes in z'.

3.3 Self-Attention

This method is an attempt to understand whether it is possible to use weights in the attention mechanism as a local interpretation of transformer architecture models. The method is based on the study [6] of the possible relationship between self-attention and feature selection methods from different points of view, including the coincidence of vocabulary, similarity of ranking, relevance of the subject area, the stability of features and the effectiveness of classification. First, for each input sequence, the average weights for the 12 heads of attention in the

last hidden layer are calculated. Next, a new matrix of weights is generated, grouping subwords in a word by averaging the weights of the subwords. The vertical average is taken as the weight of the word. Next, the top 10 words are considered as an interpretation.

This method does not depend on the response of the model, but is specific only to the transformer model. To illustrate this statement, figure 1, depicts the mean weights of the 12 self-attention heads in the last hidden state of the trained *bert-base-uncased* [2] BERT model and fine-tuned on the WOS [8] dataset for "*Interpretability of thematic classification of texts based on neural networks*". From the plot we can clearly see the so-called vertical pattern, where a few tokens receive most of the attention, such as *training, deep, transformer, language*, and *understanding*. In the work [6] they did not include special tokens $< SEP >$ and $< CLS >$ because the amount of attention received by these tokens, will make the attention received by the other tokens barely noticeable.

4 Method for Automatic Evaluation of Interpretability

After applying the interpretation methods and getting the result in a convenient form for human perception, it is necessary to understand how satisfactory the interpretation result is. As a method of evaluation, one can ask experts for assessing how clear the interpretation of the result is to them, but this is a rather resource-intensive and expensive method.

We suppose that the more explanation is similar to the category's name in the text categorization task, the more explanation is understandable for a human, This allows us to evaluate explanation methods automatically. We consider the top $N \in \{1, 3, 5, 10\}$ output words sorted by the weights assigned to them by the methods. These weights mean the importance of the word when making a decision.

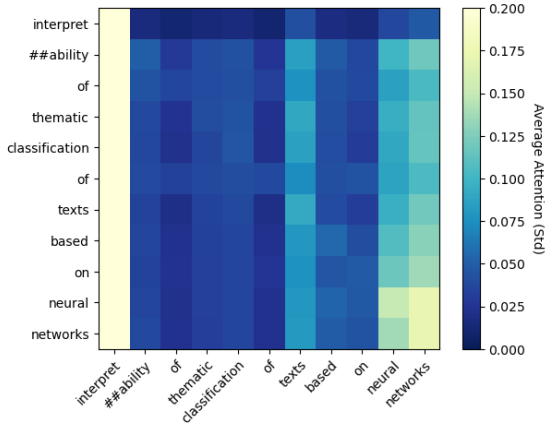

Fig. 1: An example of a matrix of weights in the form of an attention mechanism.

4.1 NDCG

Since all the above-mentioned methods of interpretation return as a result a ranking list of words with weights, we can compare these lists with the category name using the NDCG measure adapted from information retrieval [7].

Let $D^i = \{d_1^i, ..., d_m^i\}$ be the set of items retrieved for the query q^i and $\{rel^i(1), ..., rel^i(m)\}$ be their relevance labels. Let $\sigma = \{v_1, ..., v_m\}$ be the ordered items and the $DCG@k$ metric (discount cumulative gain at the position k) of the ordering is:

$$DCG@k(\sigma) = \sum_{j=1}^{k} rel(v_j)D(j), \tag{5}$$

where v_j is the identifier of the item retrieved at the position j and $D(j) = 1/log(1+j)$ is the discount function. The $NDCG@k$ metric is $NDCG@k(\sigma) = DCG@k(\sigma)/DCG@k_p$, where $DCG@k_p$ is a discount cumulative gain of the ideal ordering according to true relevance labels $rel(i)$.

In our case D^i is a list of words of the text that we interpret, q^i is the category label of the text that our deep learning model predicted, $rel(i)$ is the cosine similarity between word d_1^i and q^i embeddings. For calculation of ideal word ordering, we extract all words from the target text and arrange them in descending order of embedding similarity to the category's label: this gives us maximal DCG for the target text.

As embeddings, we use the pre-trained GloVe[1][13], fastText[2][5] and unigram_Sentence-BERT[3][14] models. The unigram prefix before Sentence-BERT means that we passed only one word to the model.

The selection of embedding models such as GloVe and fastText was made based on several factors, including their historical significance in the field of natural language processing and their widespread use as benchmarks in vari- ous NLP tasks. There are newer models like MiniLM-L12-H384 offer improvements in terms of size and quality, our decision to include these estab- lished models was driven by the desire to provide a baseline for comparison. These models serve as reference points for assessing the performance of newer approaches. We recognize the importance of incorporating the latest models and advancements in our research. In future work, we plan to expand our comparison to include more recent models.

4.2 Sentence-BERT

Sentence-BERT[14] is a modified version of the BERT network that utilizes siamese and triplet networks to generate sentence embeddings that capture semantically meaningful information. Unlike traditional methods that treat sentences as a sequence of words, Sentence-BERT takes into account the entire

[1] http://nlp.stanford.edu/data/glove.840B.300d.zip
[2] https://dl.fbaipublicfiles.com/fasttext/vectors-wiki/wiki.en.zip
[3] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

sentence as a unit during the encoding process. This allows it to capture the overall meaning and nuances of the sentence, rather than just focusing on individual words or tokens. By training the network on various tasks, such as sentence similarity or text classification, Sentence-BERT learns to produce sentence embeddings that are semantically meaningful and can be used for downstream applications like information retrieval, clustering, or sentiment analysis.

In light of the possibility of using Sentence-BERT to obtain a vector representation of sentences, we hypothesized that passing the results of interpretation methods as sentences and converting them to vector representations using Sentence-BERT could be useful. In addition, we can obtain a vector representation of a text category using Sentence-BERT and calculate the semantic proximity between the interpretation result and the category name. This will allow us to evaluate the effectiveness of the interpretation methods. For calculation the semantic similarity we use cosine similarity.

4.3 Example of Applying Interpretation And Evaluation Methods

The interpretation models were applied to a text sample from the 20NewsGroup dataset with the "baseball" label, and the model successfully predicted the category as "baseball" as well. The generated output list from the interpretation models is presented in Table 1. Each column represents the output result of interpretation, presented as a ranking list of words ordered in the rows by their weights. The NDCG@10 evaluation results of Table 1 are presented in Table 2.

The text itself: *"Pitchers are required to pitch (or feint or attempt a pick-off) within 20 seconds after receiving the ball, not 15. Pitchers are required to pitch their warm-up throws within a one minute time frame, beginning after each half inning ends, not two minutes. And the reason why a reliever should be allowed warm-ups is simple: Different mound, different catcher. Ryan Robbins Penobscot Hall University of Maine IO20456@Maine.Maine.Edu"*

In the provided example (Table 1), we observe that the SHAP method predicts a relatively general word ("relieve") as the top interpretation, while the Self_Attention method assigns a high score to the more specific term "catcher", which is highly relevant to the baseball domain. The overall NDCG@10 score for the Self_Attention method is close to the ideal value across all embeddings, indicating its strong performance. On the other hand, the SHAP method achieves a significantly lower NDCG@10 score, but when used with Sentence-BERT embeddings, it approaches the ideal value.

In terms of the evaluation using Sentence-BERT (SBERT) approach, where the interpretation results are passed as a sentence to Sentence-BERT and the semantic similarity is calculated, the results closely align with the NDCG scores, indicating a similar trend. Let SHAP sentence be "relieve, catcher, mound, pitchers, inning, ball, pitch, ups, throws, required", the LIME sentence be "inning, pitchers, not, maine, of, are, hall, is, reliever, pitch", the Self-Attention sentence be "catcher, ball, mound, pitchers, allowed, warm, off, pitch, inning, reliever" and the category sentence be "baseball".

$$
\begin{aligned}
&cos(SBERT(SHAP), SBERT(category)) = 0.56 \\
&cos(SBERT(LIME), SBERT(category)) = 0.51 \\
&cos(SBERT(Self - Attention), SBERT(category)) = 0.58
\end{aligned} \tag{6}
$$

5 Datasets

Experiments were conducted on two datasets: 20NewsGroup [1] and WOS [8].

Web Of Science (WOS) dataset is a collection of academic articles abstracts which contains three corpora (5736, 11967, and 46985 documents) for (11, 34, and 134 topics). In our work we use WOS-11967 with 34 topics. In this dataset, we have 70% for training and 30% samples are used for validation.

20NewsGroup dataset includes 18846 documents with maximum length of 1000 words. In this dataset, we have 14846 for training and 4000 samples are used for validation. We also cleaned the text from the newsgroup-related metadata contained in them, for more realistic data and so that the model does not learn to classify them. For multiword category names in *WOS* dataset (for example, *Machine learning*), we used averaging of component word embeddings for GloVe and fastText models.

Table 1: Interpretation of text about baseball from 20NewsGroup dataset.

ideal-unigram_Sentence-BERT	ideal-GloVe	ideal-fastText	SHAP	LIME	Self-Attention
pitchers	pitchers	catcher	relieve	inning	catcher
inning	catcher	pitchers	catcher	pitchers	ball
ball	inning	inning	mound	not	mound
catcher	ball	reliever	pitchers	maine	pitchers
pitch	pitch	pitch	inning	of	allowed
mound	reliever	ball	ball	are	warm
reliever	mound	mound	pitch	hall	off
throws	pick	pick	ups	is	pitch
ryan	university	university	throws	reliever	inning
hall	hall	hall	required	pitch	reliever

6 Experiments

For classification, we used *bert-base-uncased* BERT model [2] and fine-tuned it on the datasets. For 20NewsGroup we got 71.3 accuracy, and for WOS-11967 we got 86.3 accuracy.

Table 2: NDCG results for text about baseball.

	SHAP	LIME	Self-Attention
ideal-unigram_Sentence-BERT	0.93	0.75	0.90
ideal-GloVe	0.74	0.77	0.93
ideal-fastText	0.74	0.75	0.88

After we got the trained models, we used standard methods from the SHAP[1] and LIME[2] libraries. For SHAP, we used the universal *Explainer* method, which itself determined to use PartitionSHAP, faster version of KernelSHAP that hierarchically clusters features. For LIME, it was set that maximum number of features present in explanation equals 50 and size of the neighborhood to learn the linear model equals 500. For both methods we provided the label that model predicted, not the actual one. The output of interpretation modes can contain words with repetitions, so it was decided to conduct an experiment also for a unique output, when repeated words are removed from the explanation list.

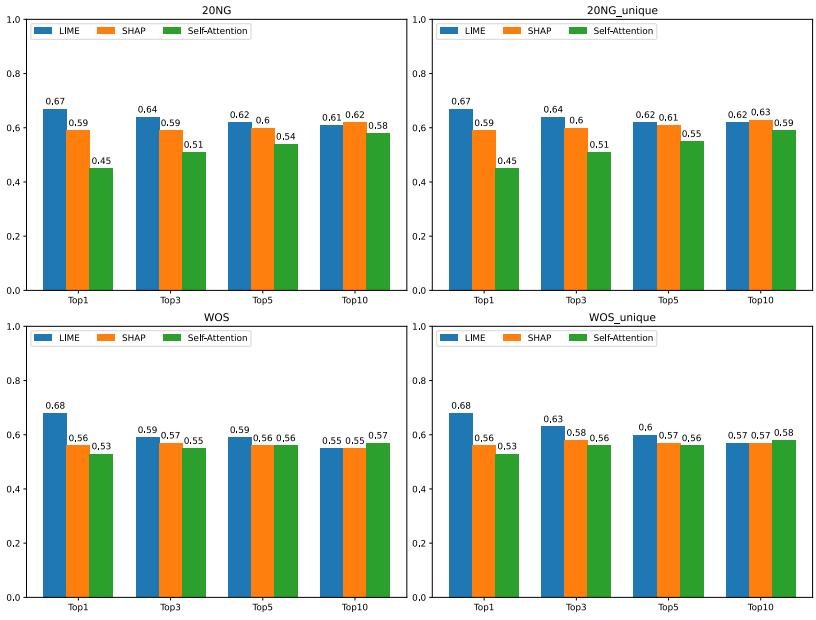

Fig. 2: Interpretation results using GloVe embeddings.

[1] https://github.com/slundberg/shap
[2] https://github.com/marcotcr/lime

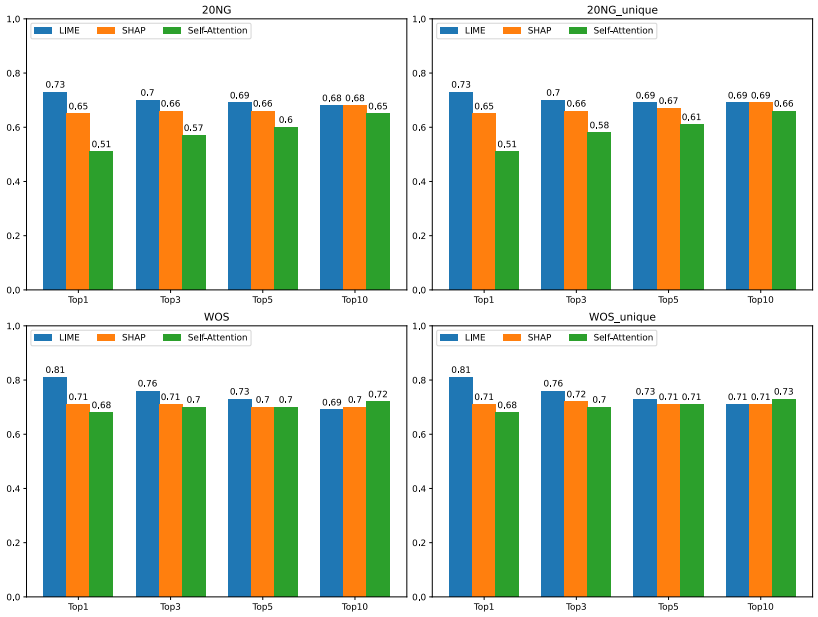

Fig. 3: Interpretation results using fastText embeddings.

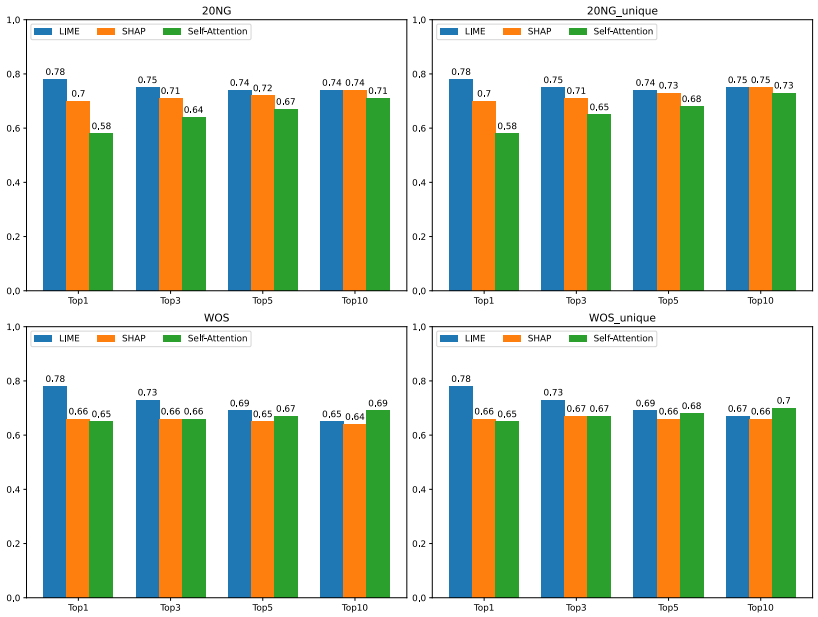

Fig. 4: Interpretation results using unigram_Sentence-BERT embeddings.

Table 3: The mean value of TopN interpretation results using NDCG evaluation.

(a) GloVe Results

	LIME	SHAP	Self-Attention
20NG	**0.630**	0.600	0.520
20NG_unique	**0.637**	0.607	0.525
WOS	**0.602**	0.560	0.552
WOS_unique	**0.620**	0.570	0.557

(b) fastText Results

	LIME	SHAP	Self-Attention
20NG	**0.700**	0.662	0.582
20NG_unique	**0.702**	0.667	0.590
WOS	**0.747**	0.705	0.700
WOS_unique	**0.752**	0.712	0.705

(c) unigram_Sentence-BERT Results

	LIME	SHAP	Self-Attention
20NG	**0.752**	0.717	0.650
20NG_unique	**0.755**	0.722	0.660
WOS	**0.712**	0.652	0.667
WOS_unique	**0.717**	0.662	0.675

Table 4: The mean value of TopN interpretation results using Sentence-BERT evaluation.

(a) Joined by spaces

	LIME	SHAP	Self-Attention
20NG	**0.352**	0.327	0.277
20NG_unique	**0.347**	0.327	0.277
WOS	**0.457**	0.437	0.385
WOS_unique	**0.447**	0.432	0.372

(b) Joined by commas

	LIME	SHAP	Self-Attention
20NG	**0.395**	0.375	0.320
20NG_unique	**0.395**	0.375	0.320
WOS	**0.472**	0.449	0.412
WOS_unique	**0.470**	0.447	0.410

The results of the experiments using NDCG evaluation are shown on Figure 2 for GloVe embeddings, Figure 3 for fastText embeddings, Figure 4 for

unigram_Sentence-Bert embeddings. The _unique postfix means that the top $N \in \{1, 3, 5, 10\}$ interpretation output contains only unique words. The Table 3 presents the results, that are averaged across various values of N, where N belongs to the set $\{1, 3, 5, 10\}$.

We can see that the first word in LIME explanations is much semantically closer to the category label for both datasets and both embeddings. LIME NDCG scores are higher at all levels than for SHAP, and almost at all levels for Self_Attention (except NDCG@10 for the WOS dataset). SHAP NDCG scores are much lower at all levels than both other scores. This agrees with findings from [9] based on human evaluation that LIME was the best method in identifying the category of an example based on explanation.

The Table 4 presents the experimental findings of the evaluation conducted on Sentence-BERT. The results are averaged across various values of N, where N belongs to the set $\{1, 3, 5, 10\}$. The suffix "_unique" signifies that the top N interpretation outputs comprise only unique words. Two approaches were explored to combine the interpretation results into sentences: the first involved joining them with spaces, while the second employed commas.

Joining by spaces involves separating words in a sentence with spaces but not adding any additional punctuation. It is a common way to represent text data in its original form for some natural language processing tasks. It is used to preserve the original structure of sentences without introducing additional separation symbols.

Joining words into sentences by commas involves using commas as delimiters between words in a sentence. This method is used for specific text analysis tasks to understand whether the presence of commas is meaningful. Maybe it can help algorithms detect relationships between items in a list or better understand sentence structure.

Comparing the two approaches revealed that using commas yielded slightly superior outcomes. The scores consistently indicate that LIME outperforms both SHAP and Self_Attention, suggesting a consistent pattern.

7 Conclusion

In this research, we proposed an automated approach for assessing the interpretability of interpretation methods in text categorization tasks. Our method involves measuring the semantic similarity between explanations and category labels using word embeddings and adapting the NDCG measure from information retrieval. Additionally, we utilized Sentence-BERT embeddings by treating the explanation as a sentence. We conducted experiments on two datasets: 20NewsGroup and the WOS dataset consisting of scientific articles. Three widely recognized interpretation methods, namely LIME, SHAP, and Self_Attention, were compared in our study. We employed GloVe, fastText, and Sentence-BERT embeddings to calculate the semantic similarity.

Our findings indicate that the LIME technique outperforms the other methods in terms of NDCG scores. Moreover, when using Sentence-BERT evaluation,

LIME consistently achieves higher scores for semantic similarity on both datasets and across all embeddings. This suggests that LIME provides more effective explanations for accurately capturing the assigned category in specific examples.

In the future, it is planned to continue the study of the interpretability of machine learning models, including trying to adjust the parameters of the LIME and SHAP methods and consider whether these results can be improved.

References

1. 20 newsgroups dataset, http://people.csail.mit.edu/jrennie/20Newsgroups/
2. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. CoRR **abs/1810.04805** (2018), http://arxiv.org/abs/1810.04805
3. Doshi-Velez, F., Kim, B.: Towards a rigorous science of interpretable machine learning. stat **1050**, 2 (2017)
4. Doumard, E., Aligon, J., Escriva, E., Excoffier, J.B., Monsarrat, P., Soulé-Dupuy, C.: A quantitative approach for the comparison of additive local explanation methods. Information Systems **114**, 102162 (2023)
5. Facebook, I.: fastText: Library for fast text representation and classification (2016), https://github.com/facebookresearch/fastText
6. Garcia-Silva, A., Gomez-Perez, J.M.: Classifying scientific publications with bert-is self-attention a feature selection method? In: Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28–April 1, 2021, Proceedings, Part I. pp. 161–175. Springer (2021)
7. Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of ir techniques. ACM Transactions on Information Systems (TOIS) **20**(4), 422–446 (2002)
8. Kowsari, K., Brown, D.E., Heidarysafa, M., Jafari Meimandi, K., , Gerber, M.S., Barnes, L.E.: Hdltex: Hierarchical deep learning for text classification. In: Machine Learning and Applications (ICMLA), 2017 16th IEEE International Conference on. IEEE (2017)
9. Lertvittayakumjorn, P., Toni, F.: Human-grounded evaluations of explanation methods for text classification. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). pp. 5195–5205 (2019)
10. Li, X., Xiong, H., Li, X., Wu, X., Zhang, X., Liu, J., Bian, J., Dou, D.: Interpretable deep learning: Interpretation, interpretability, trustworthiness, and beyond. Knowledge and Information Systems **64**(12), 3197–3234 (2022)
11. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. Advances in neural information processing systems **30** (2017)
12. Madsen, A., Reddy, S., Chandar, S.: Post-hoc interpretability for neural nlp: A survey. ACM Computing Surveys **55**(8), 1–42 (2022)
13. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). pp. 1532–1543 (2014)
14. Reimers, N., Gurevych, I.: Sentence-bert: Sentence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084 (2019)
15. Ribeiro, M.T., Singh, S., Guestrin, C.: "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. pp. 1135–1144 (2016)

16. Yalcin, M.O., Fan, X.: On evaluating correctness of explainable ai algorithms: an empirical study on local explanations for classification (2021)

8 Reviewers remarks

Review 2

Q: How do I read Table 1? Should the rows contain the same? Or should the columns in general be the same by words? Is the row order important?

A: The generated output list from the interpretation models is presented in Table 1. *Each column represents the output result of interpretation, presented as a ranking list of words ordered in the rows by their weights.*

Q: Table 3 has three subtables. It would be good to highlight the subtitles in some way.

A: Added

Q: Table 4 - the same problem with subtitles. Also, it is not clear from the text what the reason is for joining words into sentences by spaces and commas.

A: Joining by Spaces: This method likely involves separating words in a sentence with spaces but not adding any additional punctuation. It is a common way to represent text data in its original form for some natural language processing tasks. It is used to preserve the original structure of sentences without introducing additional separation symbols.

Joining by Commas: Joining words into sentences by commas involves using commas as delimiters between words in a sentence. This method is used for specific text analysis tasks to understand whether the presence of commas is meaningful. Maybe it can help algorithms detect relationships between items in a list or better understand sentence structure.

Q: Chapter 6 describes experiments and specifies datasets used. But there are no details about the number of categories for estimating NDCG. And it is unclear if anyone did an expert estimation of words within each category, at least for "ideal" cases.

A: The number of categories are provided in chapter 5 (datasets section)

Q: There are some concerns about selected embedding models. GloVe and fastText are outdated.There are much more fresh models even with small size like MiniLM-L12-H384 with better quality than an original BERT of 2018. And why there are no comparison models published after 2020?

A: The selection of embedding models such as GloVe and fastText was made based on several factors, including their historical significance in the field of natural language processing and their widespread use as benchmarks in various NLP tasks. While it's true that newer models like MiniLM-L12-H384 offer improvements in terms of size and quality, our decision to include these established models was driven by the desire to provide a baseline for comparison. These models serve as reference points for assessing the performance of newer approaches. We recognize the importance of incorporating the latest models and advancements in our research. In future work, we plan to expand our comparison to include more recent models

Q: Another concern requires additional explanation about LIME and SHAP. Both were originally used for very simple text vector models where direct interpretation is easy. Is it legitimate to use them with neural network models?

A: While LIME and SHAP were initially designed for simpler models, their application to neural network models is legitimate and valuable. In the context of complex neural networks, these methods offer a means to achieve interpretability and understand why specific predictions are made. However, it's important to acknowledge that their effectiveness may vary based on neural network architecture and complexity. Despite challenges, their use aids in model interpretability and accountability.