


RESEARCH

Open Access



IDC: quantitative evaluation benchmark of interpretation methods for deep text classification models

Mohammed Khaleel^{1*} , Lei Qi¹, Wallapak Tavanapong¹, Johnny Wong¹, Adisak Sukul¹ and David A. M. Peterson²

*Correspondence:
mkhaleel@iastate.edu
¹ Department of Computer
Science, Iowa State
University, Ames, IA 50011,
USA
Full list of author information
is available at the end of the
article

Abstract

Recent advances in deep neural networks have achieved outstanding success in natural language processing tasks. Interpretation methods that provide insight into the decision-making process of these models have received an influx of research attention because of the success and the black-box nature of the deep text classification models. Evaluation of these methods has been based on changes in classification accuracy or prediction confidence when removing important words identified by these methods. There are no measurements of the actual difference between the predicted important words and humans' interpretation of ground truth because of the lack of interpretation ground truth. A large publicly available interpretation ground truth has the potential to advance the development of interpretation methods. Manual labeling important words for each document to create a large interpretation ground truth is very time-consuming. This paper presents (1) IDC, a new benchmark for quantitative evaluation of interpretation methods for deep text classification models, and (2) evaluation of six interpretation methods using the benchmark. The IDC benchmark consists of: (1) Three methods that generate three pseudo-interpretation ground truth datasets. (2) Three performance metrics: interpretation recall, interpretation precision, and Cohen's kappa inter-agreement. Findings: IDC-generated interpretation ground truth agrees with human annotators on sampled movie reviews. IDC identifies Layer-wise Relevance Propagation and the gradient-by-input methods as the winning interpretation methods in this study.

Keywords: Machine learning interpretation, Natural language processing, Pseudo interpretation ground truth

Introduction

Deep neural networks have shown outstanding performance in various applications such as text classification [1, 2] and neural machine translation [3, 4], but they still lack interpretation transparency. Global interpretation methods reveal which parts of the neural network detect what patterns. Local interpretation methods identify parts of the input sample and their importance to the classification decision for specific input. For instance, local interpretation methods for deep learning classification models [5–17]

highlight individual words based on the relevance of the words for the predicted class label. Interpretability increases the trust in the trained models and identifies failure scenarios, thus encouraging the deployment of robust intelligent systems to end-users.

The current practice for quantitative evaluation of local interpretation methods for deep text classifiers mainly depends on utilizing variations of classification accuracy [12, 18]. For example, intrinsic validation [18] measures the decrease in classification accuracy after sequentially removing the k most important interpretation features from a given input. The greater the drop in the classification accuracy at the lowest percentage of the removed words, the better the interpretation method. The increase in confidence measure [12] is the percentage of documents in which the model's prediction confidence increases after removing the k most unimportant words from the documents.

We define the term “interpretation features” to represent words that influence the domain expert in assigning a class label to a text document. For instance, words like “fantastic,” “magical,” or “extraordinary” are example interpretation features of a positive movie review. To the best of our knowledge, there are no measurements of the actual difference between the predicted interpretation features and the ground truth. This is due to the lack of a large publicly available interpretation ground truth. We believe that such a ground truth will help advance interpretation methods by providing better quantitative evaluation. Manually creating such a ground truth is very time-consuming and prone to significant disagreement among human annotators. When our two human annotators manually marked interpretation features on the same 250 (positive and negative) product reviews, the Cohen's kappa inter-agreement [19] among their interpretation features is 0.39, far from the perfect agreement of 1. See more details in “Validation of PGTs with human annotators”.

Our contributions are summarized as follows:

- We propose IDC, a new benchmark for Interpretation methods for Deep text Classification models. IDC consists of three methods to create Pseudo interpretation Ground Truth (PGT) datasets given a labeled text classification dataset. They are gradient-based, mixed-integer linear programming, and the hybrid of the two methods. We present the inter-agreement between the PGTs and ground truth by two human annotators.
- We used our PGTs to quantitatively evaluate six recent interpretation methods, Saliency Map [10], Grad-CAM [12], DeepLIFT [14], LRP [18], Integrated Gradient [21], and Hierarchical Attention [15]. We use interpretation recall, interpretation precision, and Cohen's kappa for interpretation features as performance metrics.
- We present the first quantitative evaluation results comparing the effectiveness of these interpretation methods. The source code to generate PGTs and all the evaluated methods are available on github.com/MoKhaleel/IDC.

The remainder of this paper is as follows. “[Background on local interpretation methods](#)” section presents the background on local interpretation methods. “[Proposed IDC benchmark](#)” section details the proposed IDC benchmark, while “[Experimental design and results](#)” section presents the evaluation results of the six interpretation methods. Finally, “[Conclusion](#)” section offers a conclusion.

Background on local interpretation methods

We group the existing local interpretation methods into five categories based on how the methods work: probe internal representation [5], input perturbation [6, 7], signal backpropagation [8, 9], gradient-based [10–14], and attention-based approaches [15–17]. Many of these methods were first introduced as interpretation methods for image classification models, and some were later applied for text classification models.

Probe internal representation

This approach examines the input and output signal of a particular neuron of a neural network layer. For instance, the method by Jacovi et al. [5] identifies the relevant n -grams of words that contribute to the class prediction by examining the activation score of each convolutional filter. Then, the contribution of each word in the relevant filter is calculated to get the word relevance score. They evaluated the importance of relevant words by measuring the drop in text classification accuracy without using the unimportant convolutional filters that produce activation scores below a predefined threshold.

Input perturbation approach

This approach removes parts of the input text such as words, phrases, or sentences. It calculates the change in the classification probability to measure the importance of the removed element. The Leave-one-out method [7] removes one word at a time from the input text and measures the percentage of reduction in classification confidence as an importance measure. Leave-one-out suffers from computational overhead, especially with long input text. Furthermore, it does not consider dependency among words. That is, removing one word may not drop the classification confidence, but removing a small subset of words does. LIME [6] solves this problem by training a linear model to select a predetermined subset of words that strongly affect the classification confidence.

Signal backpropagation approach

This approach assigns a relevance score to each word in the input by redistributing the output score of the predicted class via backpropagation to each neuron in the previous layer. It repeats all the way to the input layer. Layer-wise Relevance Propagation (LRP) [9, 18] was applied to interpret text classification models on a topic categorization task. As opposed to interpretation methods, LRP can distinguish the words that positively and negatively contribute to the classification decision. However, LRP is computationally expensive since it calculates the relevance score for every neuron in every layer [14]. Class Activation Map (CAM) [8] eliminates the computational overhead by multiplying the weight connections of the predicted class with the corresponding feature maps of the last convolutional layer and then up-samples the generated heatmap of relevance scores. Intrinsic validation was used to evaluate the performance of LRP and CAM.

Gradient-based approach

This approach relies on gradients to lead to the input parts that are important for the classification decision of a given input. The higher the gradient value, the more the contribution. Starting backward from the output layer, the gradient with respect to the predicted class is backpropagated to some desired layer, and the gradient-related scores are

distributed to the input text. This generates a gradient map, called the Saliency Map [10]. Gradient-Class Activation Map (Grad-CAM) [12] multiplies the gradient of the predicted class with the values of the feature maps output of the last convolutional layer (gradient-by-input). Grad-CAM redistributes the multiplication results backward to the input layer via up-sampling. Grad-CAM works with a wide variety of CNN-based models without changing the underlying network architecture. These methods, however, suffer from the gradient saturation and gradient threshold problems, which have been addressed by DeepLIFT [14] and Integrated Gradient [21]. The methods of this approach use the intrinsic validation and the increase in confidence as evaluation metrics.

Attention-based approach

Attention-based classification methods have become popular due to their ability to improve classification accuracy. During training, the classifier learns the weights of an attention vector to emphasize the relevant parts of the input for classification. The learned attention vector is then used to locate the relevant words in the input [16, 17] as interpretation. However, recent study shows that the interpretation generated by attention methods is not accurate [22, 23].

Proposed IDC benchmark

To the best of our knowledge, there are no large publicly available datasets with labeled interpretation features. Our experience indicates that manual labeling of interpretation features at the word level is much more time-consuming than deciding the class labels of documents. The process is also prone to more disagreement among the human annotators in judging which words are relevant to the class label, especially for long documents and datasets with several classes. Both obstacles hinder the creation of a large interpretation ground truth. Human-annotated datasets like ERASER [20] and E-SNLI [24] provide the annotation only at a clauses or phrases level. Therefore, they do not precisely measure the word-level performance of the interpretation methods.

We propose three methods to generate PGT—a pseudo ground truth of interpretation features from a text classification dataset with a class label per document. We describe the performance metrics on PGTs for the evaluation of interpretation methods. Our methods use the following notations. Let C be a set of class labels with $|C|$ denoting the cardinality of the set. Let the ground truth classification dataset $D = (X, Y)$ where $X = \{x_1, \dots, x_m\}$; $Y = \{y_1, \dots, y_m\}$; m is the number of documents; x_i represents a document i , and $y_i \in C$ is the corresponding class label of x_i .

Method 1: weighted gram activation (WGA)

WGA is inspired by the gradient-based interpretation approach that scores the input words based on their relevance to the classification decision. Figure 1 summarizes the algorithm. In Line 1, `ExtractUniqueTerms(X)` returns \mathcal{V} —a set of unique words extracted from X . In Line 2, the WGA function returns a 2-dimensional matrix RS where each element $RS[w, c]$ is the average relevance score of the word w in all documents correctly classified as class c . The WGA function will be explained in more detail shortly. In Lines 3–9, the relevance scores in RS are used to assign each unique word as an interpretation feature for at most one class. Specifically, Line 5 gets the relevance score for the word w

```

Input:
D: Classification dataset;  $D = (X, Y)$ 
M: CNN classification model trained on D
 $\alpha$ : Relevance threshold
 $\beta$ : Margin difference

Output:
 $IF_c$ : Interpretation features set for class c
 $IF_c$  is initially an empty set.

Algorithm:
1:  $\mathcal{V} = \text{ExtractUniqueTerms}(X)$ 
2:  $RS = \text{WGA}(M, D, \mathcal{V})$  #compute relevance scores
   # post-processing
3: for each word  $w \in \mathcal{V}$  do
4:   for each class  $c \in C$  do
5:      $Relv = RS[w, c]$ 
6:     if  $Relv \geq \alpha$  do
7:       for  $\hat{c} \in C$  and  $\hat{c} \neq c$  do
8:         if  $Relv - RS[w, \hat{c}] \geq \beta$  do
9:           Append  $w$  to  $IF_c$ 

```

Fig. 1 Weighted gram activation algorithm

and the class c . Lines 6–9 check whether this word can be assigned to the class c . That is, the relevance score of the word w for the class c is at least the relevance threshold α and larger than its relevance score for other classes by at least β . Both α and β are constant values between 0 and 1.

The WGA function takes the pre-trained model M , the training dataset D , and \mathcal{V} , respectively. The model M predicts the class label of each training document. Any CNN-based classifiers that offer high classification accuracy can be used. Our implementation used a well-known CNN model for text classification [25] because CNN models and LSTM models do not differ much in their performance [16, 26]. The CNN model [25] uses one CNN layer with three window sizes covering 3, 4, and 5 words. The model M runs through all the documents in D , and in the next steps only considers the correctly predicted. For each such document, a single backward pass generates scores for all of the words based on their relevance to the correct classification of the document. We adapt the gradient-by-input idea for computing relevance scores. The gradient-by-input idea was used in several recent gradient-based interpretation methods for image classification.

Let's consider a training document d with n words, which is correctly predicted to be of class c . During the prediction, the q convolutional filters f_1, \dots, f_q with a window size of r -grams generate (using padding) a corresponding feature map $FM \in \mathbb{R}^{n \times q}$, as shown in Fig. 2. Let $GM^c \in \mathbb{R}^{n \times q}$ be a matrix of the gradients with respect to the class c . The \max function finds the maximum gradient for each row in GM^c and outputs a gradient vector $\delta^c \in \mathbb{R}^n$, where each element represents the strongest gradient for each r -gram in the document d . The next step generates the vector $z^c \in \mathbb{R}^n$ with one relevance score for each of the r -grams using Eq. (1) to calculate each score value $z^c[j]$. First, it calculates the sum of the product of the highest gradient of the r -grams j and the feature value from the convolution of each filter k to that r -grams. The ReLU function suppresses any negative values in the sum. Note that padding is used to keep the number of the r -grams the same for all window sizes.

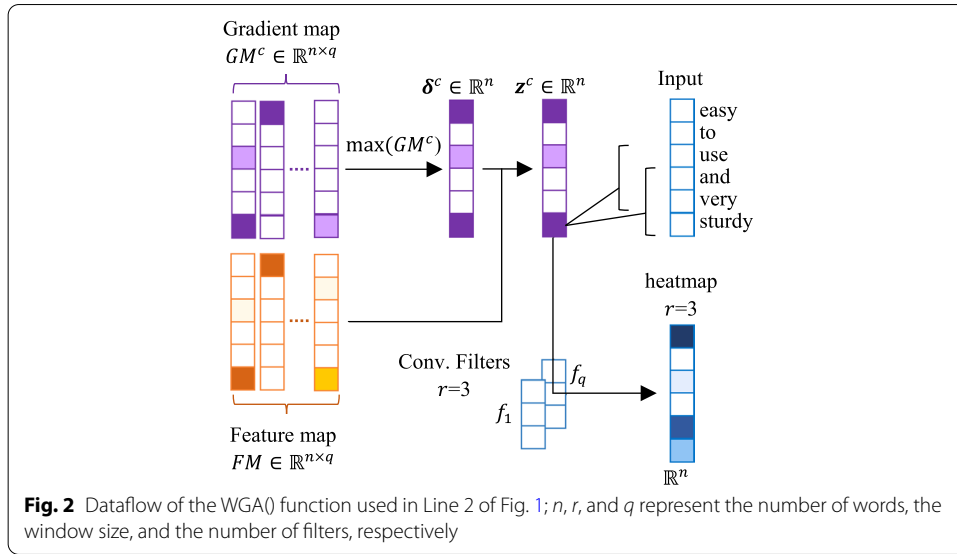


Fig. 2 Dataflow of the WGA() function used in Line 2 of Fig. 1; n , r , and q represent the number of words, the window size, and the number of filters, respectively

$$z^c[j] = \text{ReLU} \left(\sum_{k=1}^q (\delta^c[j] \times FM[j, k]) \right), \tag{1}$$

$$\forall j = 1 \dots n \text{ and } \delta^c = \max_q (GM^c)$$

Adapting the idea for text classification is not straightforward. Jacovi et al. found that different words in one convolutional filter of r -grams do not necessarily contribute equally to generating the feature value [5]. For instance, in Fig. 2, the input at position 5 ($j = 5$) which corresponds to the word “very” is part of the 3-g filter “use and very”. The words “use”, “and”, and “very” in this gram do not necessarily contribute equally to the feature value at this position z^c [5]. Furthermore, each filter of size 3 words processes the word “very” twice in the “use and very” and “and very sturdy” grams to calculate the feature value at this position. Therefore, the relevance score for the word “very” should be normalized based on the filter weight corresponding to this word in all three-word grams for this filter. Equation (2) is used to distribute the r -gram relevance score of $z^c[j]$ to each word w in the r -gram that includes the word w based on the relevant filter weights of a convolutional filter f .

$$s^w(f) = \frac{e(w) \cdot f[idx(w)]}{Conv([\dots, w, \dots], f)} \times z^c[j], \tag{2}$$

where $e(w)$ is the word embedding of w ; $idx(w)$ is the index of the convolutional filter f weight that corresponds to the word w ; \cdot is the dot product; and $Conv()$ is the convolutional operation of the r -gram of words corresponding to the convolutional filter f . The $s^w(f)$ function calculates the relevance score of the word w for each filter f with the window size r . Thus, it obtains one heatmap of relevance scores for each window size r . Finally, the average of the relevance scores at the same position across all the heatmaps is computed to get the final heatmap for this document. The average of all the

relevance scores of the word w in all documents of class c is stored in the matrix element $RS[w, c] \in [0..1]$. As shown in Fire 2, the more saturated the cell, the more important it is to the classification decison.

Method 2: mixed integer linear program (MILP)

The previous method pends on the effectiveness of the chosen text classifier to generate relevance scores for classification. Our second method does not use classifiers but analyzes word distribution in each class. The first step constructs the set of unique words \mathcal{V} from the training dataset by selecting $|\mathcal{V}|$ words with the highest Term-Frequency-Inverse-Document-Frequency (TF-IDF) values [34]. Given \mathcal{V} , we want to (i) find the smallest set of interpretation features for each class, (ii) restrict the interpretation feature sets of all the classes to be mutually exclusive, and (iii) restrict the chosen interpretation features for a class to occur in all the documents of that class and as few as possible of the documents of the other classes. We adapt the formulation of a constrained mixed-integer linear program in [27] to achieve the above goals.

Let IF_c denote a set of interpretation features for class c and $\mathbf{v}^c \in \mathbb{R}^{|\mathcal{V}|}$ be the one-hot encoding of IF_c . such that when the j th position of \mathbf{v}^c is set to 1, the j th word is selected as an interpretation feature for the class c . The optimization goal is to minimize Eq. (3) given the parameter set $\theta = \{\mathbf{v}^c, \forall c \in C\}$. Equation (3) means that each interpretation feature set IF_c has the fewest words that appear in the documents of the class c while occurring the fewest times in the documents of the other classes $\forall \hat{c} \in C$ and $\hat{c} \neq c$. Equation (4) restricts each word to be selected as an interpretation feature of at most one class. Equation (5) ensures that the set of words chosen as interpretation features of a class appears in all the training documents of the class.

$$\text{minimize}_{\theta} \sum_{c \in C} \sum_{\mathbf{d} \in D^c} \sum_{\hat{c} \in C, \hat{c} \neq c, \mathbf{d} \cdot \mathbf{v}^{\hat{c}}}, \tag{3}$$

subject to:

$$\sum_{c \in C} \mathbf{v}^c[j] \leq 1, \forall j = 1..|\mathcal{V}| \tag{4}$$

$$\mathbf{d} \cdot \mathbf{v}^c \geq 1, \forall c \in C, \forall \mathbf{d} \in D^c, \tag{5}$$

where $\mathbf{d} \in D^c$ denotes a row of the matrix D^c , which is a one-hot vector representation of size $|\mathcal{V}|$ of a training document in the class c ; D^c has as many rows as the number of training documents in class c .

Method 3: hybrid

We designed this method to achieve the benefits of both WGA and MILP. The first two steps of Hybrid use Lines 1–2 of the WGA algorithm in Fig. 1 to construct the set of unique words \mathcal{V} as the words with the highest relevance scores. Then, the mixed-integer linear programming is applied as done in Method 2. The goal of using the WGA function is that the word relevance scores have a higher correlation to the target class than the TF-IDF values.

Interpretation performance metrics

Given PGTs, we can, for the first time in the literature, quantify the effectiveness of interpretation methods without relying on classification accuracy. We adapt the widely used metrics: Cohen's kappa, precision, and recall based on interpretation features as the performance metrics. Cohen's kappa agreement accounts for the possibility of the agreement occurring by chance and disagreement between raters [31, 32]. Equations (6) and (7) define the interpretation precision $P_{interp.}$ and interpretation recall $R_{interp.}$, respectively.

$$P_{interp.} = \frac{tp}{tp + fp} \quad (6)$$

and

$$R_{interp.} = \frac{tp}{tp + fn}, \quad (7)$$

where tp is the number of correctly labeled interpretation features by an interpretation method; fp is the number of incorrectly labeled interpretation features, and fn is the number of the interpretation features that are incorrectly labeled as unimportant words. The interpretation recall indicates the effectiveness in finding correct interpretation features. The interpretation precision indicates the effectiveness in lowering false interpretation features.

Experimental design and results

We designed our experiments to address the following research questions: (1) How close are the generated PGTs to human reasoning? (2) What is the performance of the interpretation methods when evaluated against the PGTs? And (3) Does removing the PGTs from the text documents impact classification confidence? We answer these questions in Sects. 4.2–4.4, respectively.

Classification models and hyperparameters settings

All experiments used the CNN text classifier with one convolutional layer and three window sizes covering 3, 4, and 5 words [25]. The CNN classifier of [25] was chosen due to its simplicity and good performance for the text classification task. A total of 50 convolutional filters for each window size was found sufficient enough to capture the important words [5]. Each word was represented as a 128-dimension word2vec embedding vector initialized randomly and then optimized during the training. Padding was used to handle the varying document lengths. The training dropout rate was empirically set to 0.5, and the minibatch size was 32. The Adam optimizer with a learning rate of 0.001 was used. The training process lasted at most ten epochs for each dataset. For WGA, the threshold α was 0.3 and the threshold β was 0.25 for all the datasets used in our experiments. Our implementation was done using Python 3 with TensorFlow 2.5.

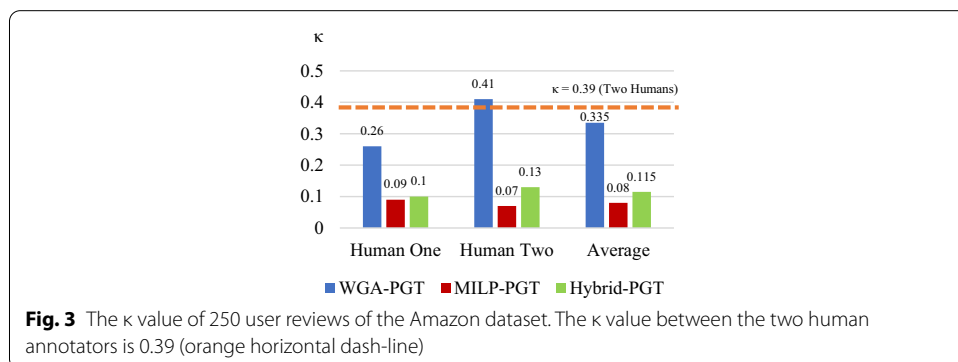
Validation of PGTs with human annotators

We compared our PGTs with those generated by two human annotators using 125 negative reviews (rated as 1 star and 2 stars) and 125 positive reviews (rated as 4 stars and 5 stars). These reviews were randomly sampled without replacement from the Amazon product reviews [28]. Consistent with existing work, The three-star user reviews were not considered [25]. It is difficult to manually label interpretation ground truth for each rating using the 5-star rating system. We asked two native English speakers with a university level education to individually label the same reviews as negative or positive. We provided the two annotators with guidelines on labeling to reduce the disagreement. The guidelines include ignoring the articles, focusing on active verbs, and considering each word as independent and identically distributed. With these guidelines, the Cohen’s kappa κ [19] inter-agreement between the two human annotators is 0.39. Note that the kappa κ value is between -1 and 1 . The positive value indicates the agreement between the two raters. The two humans agreed on some key interpretation features, but several words were considered important by one person but not the other. A good PGT should overlap with the commonly agreed interpretation features.

To compare the κ inter-agreement of our PGTs and the ground truth by the annotators, one set of PGTs was generated for each of the three proposed methods. The generated PGTs were obtained from the same Amazon reviews given to the human annotators. Figure 3 shows the κ inter-agreement value between each PGT and each human annotator.

Due to the lack of MILP libraries that utilize parallel computation, it is infeasible to obtain an optimal solution for a very large dataset using the mixed-integer linear programming solver. Our implementation uses the Solving Constraint Integer Programs [29]. For a large training dataset like Amazon, 5 K reviews were randomly selected from the training dataset to construct the PGTs with the MILP and Hybrid methods with a vocabulary size $|\mathcal{V}|$ of 2000 words.

The WGA-PGT outperforms the MILP-PGT and the Hybrid-PGT in terms of κ values. The agreement of WGA-PGT and the first human expert is lower than that of the two humans. Recall that the two experts were provided with guidelines on labeling the interpretation features to reach a high inter-agreement. However, the agreement of the WGA-PGT and the second human expert outperforms the agreement of the two human annotators. This reflects the high confidence of our PGT. The performance of the



MILP-PGT and the Hybrid-PGT would gain high improvement once trained on a larger dataset. This would be possible by getting MILP libraries that utilize parallel computational power. Figure 4 shows the 20 most relevant interpretation features from WGA-PGT for the positive and negative Amazon reviews.

Evaluation of existing interpretation methods for text classification

We compared interpretation effectiveness of the state-of-the-art local text interpretation methods: saliency Map [10], Grad-CAM [12], Integrated Gradient [21], DeepLIFT [14], LRP [18], and Hierarchical Attention [10]. The word-level attention layer was used for the Hierarchical Attention to generate the word-relevant scores. The authors’ original code was used in our experiments.

To evaluate the interpretation method against the generated PGTs, the CNN classifier for sentiment analysis (positive and negative) was trained on the Movie Review polarity (MR) dataset [30]. The MR dataset contains 5331 positive and 5331 negative movie reviews [33] and is commonly used in several machine learning interpretation for text classification [5]. The dataset was split into 60%, 20%, and 20% for training, validation, and testing, respectively. We used our proposed methods to generate PGTs since no existing methods were proposed to generate PGTs for interpretation. The PGTs were generated using the training dataset. The number of unique words $|\mathcal{V}|$ was 3000 words for the MILP and Hybrid methods. To reach the optimal solution for both MILP and Hybrid methods, the documents with less than five words were discarded from the construction of the vocabulary. As a result, the number of training documents used to generate the PGTs was 2 K for the MILP method and 7 K for the Hybrid method.

Interpretation effectiveness

Table 1 shows the interpretation effectiveness using the kappa, interpretation precision, and interpretation recall on the MR dataset. The interpretation method (listed as the column name in Table 1) with the highest κ , P_{interp} , and R_{interp} values (with a bold font format) is best at interpretation. All PGTs agree that LRP and the gradient-by-input methods outperform Saliency Map and Hierarchical Attention. Even with 25% of the training dataset, the interpretation performance of MILP is about half of that of WGA. Hence, this method can perform better when trained with a full-size dataset and large vocabulary size. The Hybrid method does not perform as well as expected due to the

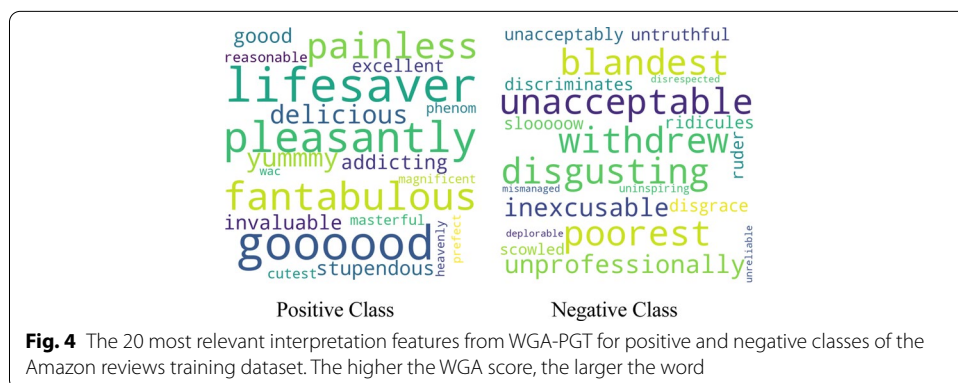


Table 1 The interpretation effectiveness of the interpretation methods using different PGTs on the MR test dataset. The number under each PGT indicates the documents in the training dataset used to generate that PGT. The bold font format indicates the highest score

PGT	Performance metric	Interpretation methods					
		Saliency map	Grad-CAM	Integrated gradient	DeepLIFT	LRP	Hierarchical attention
WGA (8 K)	κ	0.33	0.58	0.58	0.58	0.59	0.16
	P_{interp}	0.40	0.71	0.70	0.72	0.72	0.28
	R_{interp}	0.53	0.61	0.61	0.62	0.63	0.29
MILP (2 K)	κ	0.17	0.25	0.25	0.25	0.25	0.09
	P_{interp}	0.21	0.29	0.29	0.30	0.29	0.13
	R_{interp}	0.26	0.30	0.30	0.30	0.31	0.19
Hybrid (7 K)	κ	0.27	0.43	0.42	0.43	0.44	0.15
	P_{interp}	0.41	0.57	0.56	0.58	0.58	0.27
	R_{interp}	0.36	0.52	0.50	0.51	0.53	0.28

limitation of mixed-integer linear programming. LRP achieves the best performance in terms of the kappa, interpretation precision, and interpretation recall. However, the difference in the results with Grad-CAM, DeepLIFT, and Integrated Gradients is marginal. Hierarchical Attention has the lowest performance. This finding concurs with [22, 23]. Therefore, we conclude that these methods are comparable.

Impact of interpretation features on classification confidence

Does removing the PGTs interpretation features from the text impact classification confidence? To answer this question, Eq. (8) was used to calculate the average relative classification confidence after removing all the interpretation features of the target class from the documents of the MR test dataset. The more the drop in relative confidence at the smallest subset of words, the better the PGT method is. The relative confidence score value is from zero to one.

$$s^w(f) = \frac{e(w) \cdot f[idx(w)]}{Conv([..., w, ...], f)} \times z^c[j],$$

$$RelativeConf = \frac{1}{n} \sum_d \left(y_d - \frac{\max(0, y_d - \hat{y}_d)}{y_d} \right), \tag{8}$$

where y_d is the confidence score of the correctly classified test document d and \hat{y}_d is the confidence score for the same predicted class with some words removed from d ; n is the total number of documents.

The maximum relative confidence of the CNN text classifier is defined as the relative confidence using the entire text without removing any words. For the MR test dataset, the maximum relative confidence is 0.75. WGA achieves the best relative confidence of 0.52, a drop of 0.23 from the maximum. Hybrid achieves the second-best relative confidence of 0.61. MILP performs the worst with a relative confidence of 0.57, a drop of 0.17 compared with the maximum. We conclude that these interpretation

features significantly impact the text classifier confidence and thus are important for classification.

Conclusion

This paper presents an alternative approach for quantitative evaluation of interpretation methods for deep text classifiers. Unlike prior work, the proposed approach does not rely on classification accuracy or prediction confidence. Most local interpretation methods indicate important words (interpretation features) used by the classifier to predict the class label for a given document. If the interpretation features match well with what a user expects, the user's trust in the model increases. Given that automated decision systems are projected to be prevalent in daily lives, it is vital to improving interpretation methods. A good benchmark is essential for such an effort.

The proposed IDC benchmark consists of (1) three methods (WGA, MILP, and Hybrid) for generation of the pseudo interpretation ground truth and (2) performance metrics, namely, interpretation recall, precision, and Cohen's kappa inter-agreement. These metrics are adapted for interpretation features from commonly used performance metrics for other tasks. WGA and MILP are significantly different in their design, but they indicate the same finding of which interpretation methods are best. Using the IDC benchmark, the best interpretation methods in this study are LRP and the gradient-by-input methods followed by Saliency Map and Hierarchical Attention. LRP gives the interpretation that matches better with the pseudo interpretation ground truth. Interpretation features derived from attention vectors do not match well with the pseudo interpretation ground truth. Our result confirms a similar finding by other studies.

WGA outperforms MILP and Hybrid in terms of its ability to process a large dataset and give the pseudo interpretation ground truth that better matches those of the human annotators. The pseudo ground truth by WGA has the highest inter-agreement rating with the two human annotators on product reviews of 250 documents. The study provides supportive evidence that a very high agreement at word levels among the human annotators is challenging to achieve. Identifying important phrases in a document is more feasible to obtain a higher agreement among human annotators. WGA is extendable to generate phrase-level interpretation ground truth from word-level interpretation features and their importance scores. The question of what the best interpretation is for the users of automated decision support systems is beyond the scope of the paper. The answer would benefit from more collaborative work of cognitive scientists, computational linguists, and computer scientists.

Acknowledgements

Not applicable.

Authors' contributions

MK and WT proposed the idea, designed the algorithm and the experiments, and wrote the manuscript. In addition, MK implemented the code, gathered the experimental data, and collected the results. All authors discussed the idea and contributed to the proposed work. All authors read and approved the final manuscript.

Funding

This work is partially supported by NFS Grant No. 1729775. Findings, opinions, and conclusions expressed in this paper do not necessarily reflect the view of the funding agency.

Availability of data and materials

The datasets generated during the current study are available from the corresponding author on reasonable request. We provide the source code to generate PGTs and all the methods we evaluated on <https://github.com/MoKhaleel/IDC>.

Declarations

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Author details

¹Department of Computer Science, Iowa State University, Ames, IA 50011, USA. ²Department of Political Science, Iowa State University, Ames, IA 50011, USA.

Received: 8 October 2021 Accepted: 18 February 2022

Published online: 26 March 2022

References

1. He Y, Li J, Song Y, He M, Peng H. Time-evolving text classification with deep neural networks. In: the international joint conference on artificial intelligence. 2018. p. 2241–7.
2. Yuan Z, Wu F, Liu J, Wu C, Huang Y, Xie X. Neural sentence-level sentiment classification with heterogeneous supervision. In: the IEEE international conference on data mining. 2018. p. 1410–5.
3. Luong M-T, Pham H, D. Manning C. Effective approaches to attention-based neural machine translation. In: the conference on empirical methods in natural language processing. 2015. p. 1412–21.
4. Wu Y, Schuster M, Chen Z, Le Q V, Norouzi M, Macherey W, et al. Google's neural machine translation system: bridging the gap between human and machine translation. ArXiv. 2016. p. 1–23.
5. Jacovi A, Sar Shalom O, Goldberg Y. Understanding convolutional neural networks for text classification. In: continual rare-class recognition with emerging novel subclasses in joint European conference on machine learning and knowledge discovery in databases. 2019. p. 20–36.
6. Ribeiro MT, Singh S, Guestrin C. Why should I trust you? Explaining the predictions of any classifier. In: The ACM SIGKDD international conference on knowledge discovery and data mining. 2016. p. 1135–44.
7. Li J, Monroe W, Jurafsky D. Understanding neural networks through representation erasure. ArXiv. 2016. p. 1–16.
8. Zhou B, Khosla A, Lapedriza A, Oliva A, Torralba A. Learning deep features for discriminative localization. In: IEEE conference on computer vision and pattern recognition. 2015. p. 2921–9.
9. Bach S, Binder A, Montavon G, Klauschen F, Müller KR, Samek W. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *Public Libr Sci PLoS ONE*. 2015;10(7):1–46.
10. Denil M, Demiraj A, Freitas N De, Kingdom U, Deepmind G. Extraction of salient sentences from labelled documents. *Tech Rep Univ Oxford*. 2015;1–9.
11. Simonyan K, Vedaldi A, Zisserman A. Deep inside convolutional networks: visualising image classification models and saliency maps. In: the international conference on learning representations. 2014. p. 1–8.
12. Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D. Grad-CAM: visual explanations from deep networks via gradient-based localization. In: the IEEE international conference on computer vision. 2017. p. 618–26.
13. Chattopadhyay A, Sarkar A, Howlader P. Grad-CAM ++: generalized gradient-based visual explanations for deep convolutional networks. In: the IEEE winter conference on applications of computer vision. 2018. p. 839–47.
14. Shrikumar A, Greenside P, Kundaje A. Learning important features through propagating activation differences. In: the international conference on machine learning. 2017. p. 4844–66.
15. Yang Z, Yang D, Dyer C, He X, Smola A, Hovy E. Hierarchical attention networks for document classification. In: the conference of the North American chapter of the association for computational linguistics: human language technologies. 2016. p. 1480–9.
16. Wang S, Huang M, Deng Z. Densely connected CNN with multi-scale feature attention for text classification. In: the international joint conference on artificial intelligence. 2018. p. 4468–74.
17. Poria S, Cambria E, Hazarika D, Mazumder N, Zadeh A, Morency L-P. Multi-level multiple attentions for contextual multimodal sentiment analysis. In: the IEEE international conference on data mining. 2017. p. 1033–8.
18. Arras L, Horn F. "What is relevant in a text document?": an interpretable machine learning approach. *Public Libr Sci PLoS ONE*. 2017;8(12):1–24.
19. Vieira SM, Kaymak U, Sousa JM. Cohen's kappa coefficient as a performance measure for feature selection. In: the IEEE international conference on fuzzy systems. 2010. p. 1–8.
20. DeYoung J, Jain S, Rajani NF, Lehman E, Xiong C, Socher R, et al. ERASER: a benchmark to evaluate rationalized NLP models. In: annual meeting of the association for computational linguistics. 2020.
21. Sundararajan M, Taly A, Yan Q. Gradients of counterfactuals. ArXiv. 2016. p. 1–19.
22. Serrano S, Smith NA. Is attention interpretable? arXiv. 2019;2931–51.
23. Wiegrefse S, Pinter Y. Attention is not explanation. In: the conference of the North American chapter of the association for computational linguistics: human language technologies. 2019. p. 11–20.
24. Camburu O-M, Rocktäschel T, Lukasiewicz T, Blunsom P. e-SNLI: natural language inference with natural language explanations. In: conference on neural information processing systems. 2018.
25. Kim Y. Convolutional neural networks for sentence classification. In: the conference on empirical methods in natural language processing. 2014. p. 1746–51.

26. Wang J, Yu L-C, Lai KR, Zhang X. Dimensional sentiment analysis using a regional CNN-LSTM Model. In: proceedings of the 54th annual meeting of the association for computational linguistics. 2016. p. 225–30.
27. Nguyen H, Wang X, Akoglu L. Continual rare-class recognition with emerging novel subclasses. *lecture notes computer science artificial intelligent and bioinformatic*. 2020. p. 20–36.
28. He R, McAuley J. Ups and downs: modeling the visual evolution of fashion trends with one-class collaborative filtering. In: the 25th international conference on world wide web. 2016. p. 507–17.
29. Achterberg T. SCIP: solving constraint integer programs. *Math Progr Computer*. 2009;1(1):1–41.
30. Pang B, Lee L. Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales. In: the annual meeting of the association for computational linguistics (ACL). 2005. p. 115–24.
31. McHugh M. Interrater reliability: the kappa statistic. *Biochemia medica*. 2012;22:276–82.
32. Gerald R, Shih Y. Evaluation of Cohen's kappa and other measures of inter-rater agreement for genre analysis and other nominal data. *Journal of english for academic purposes*. 2021; 53.
33. Pang B, Lee L: A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts. In: proceedings of ACL. 2004.
34. Rajaraman A, Jeffrey D. *Data Mining, Mining of massive datasets*. Cambridge: Cambridge University Press; 2011. p. 1–17.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Submit your manuscript to a SpringerOpen[®] journal and benefit from:

- ▶ Convenient online submission
- ▶ Rigorous peer review
- ▶ Open access: articles freely available online
- ▶ High visibility within the field
- ▶ Retaining the copyright to your article

Submit your next manuscript at ▶ [springeropen.com](https://www.springeropen.com)
