IDC: Quantitative Evaluation Benchmark of Interpretation Methods for Deep Text Classification Models

Mohammed Khaleel (mkhaleel@iastate.edu)
Iowa State University
Lei Qi
Iowa State University
Wallapak Tavanapong
Iowa State University
Johnny Wong
Iowa State University
Adisak Sukul
Iowa State University
David Peterson
Iowa State University

Research

Keywords: achine Learning Interpretation, Natural Language Processing, Pseudo Interpretation Ground Truth

Posted Date: October 29th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-960359/v1

IDC: Quantitative Evaluation Benchmark of Interpretation Methods for Deep Text Classification Models

Mohammed Khaleel[1*], Lei Qi[1], Wallapak Tavanapong[1], Johnny Wong[1], Adisak Sukul[1], David Peterson[2]

[*] Corresponding author
[1] Department of Computer Science, Iowa State University, IA 50011, USA
[2] Department of Political Science, Iowa State University, IA 50011, USA
{mkhaleel, leiqi, tavanapo, wong, adisak, daveamp}@iastate.edu

Abstract

Recent advances in deep neural networks have achieved outstanding success in natural language processing tasks. Due to the success and the black-box nature of the deep text classification models, interpretation methods that provide insight into the decision-making process of these models have received an influx of research attention. The evaluation of these methods is typically done by observing classification accuracy or prediction confidence when important word grams are removed. Due to the lack of interpretation ground truth, there are no measurements of the actual difference between the predicted important words and the interpretation ground truth. Manual labeling of a large interpretation ground truth is time-consuming. We propose a new benchmark for quantitative evaluation of **I**nterpretation methods for **D**eep text **C**lassification (IDC) models. The IDC benchmark consists of the following. 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. We used our benchmark to evaluate six state-of-the-art interpretation methods.

Keywords: *Machine Learning Interpretation; Natural Language Processing; Pseudo Interpretation Ground Truth.*

1 Introduction

Deep neural networks have shown outstanding performance in a wide range of 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 part(s) of the input sample and their importance to the classification decision for a specific input. Machine learning interpretation of natural language processing tasks is still in its infancy. We believe it would take experts in cognitive science, linguistics, and computer science together to provide desirable and computable properties of good interpretation. In this paper, we focus on local

interpretation of text classification tasks. We categorize the local interpretation methods into five categories: Probe internal representation [5] , perturbation [6, 7], signal backpropagation [8, 9], gradient-based [10, 11, 12, 13, 14], and attention-based approaches [15, 16, 17]. These methods highlight individual words based on relevance of the words for the predicted class label.

The current practice for quantitative evaluation of existing interpretation methods mainly depends on utilizing variations of classification accuracy [12, 18]. For example, intrinsic validation [18] is a measure of the decrease in classification accuracy after sequentially removing k most important interpretation features from a given input. The more drop in the classification accuracy at the lowest percentage of the removed words, the better the interpretation method. The increase in confidence measure is the percentage of documents in which the model's prediction confidence increases after removing k most unimportant words from the documents [12]. 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 as in quantitative evaluation of text classifiers. This is due to the lack of a large publicly available interpretation ground truth. We believe that a large public interpretation ground truth will help advance interpretation methods by providing better quantitative evaluation metrics than classification-based metrics. Manually creating such a ground truth is very time consuming and prone to significant disagreement among human annotators. In fact, we had two human annotators manually mark interpretation features on the same 250 (positive and negative) product reviews. Even with this small review dataset that only English fluency is required for annotation, the Cohen's kappa inter-agreement [19] among the interpretation features marked by the two annotators is 0.39, far from the perfect agreement of 1. See more details in Section 4.2. Although there is a human annotated annotation of rationales---spans of words, clauses, or phrases [20], the rationale level ground truth cannot precisely measures word-level performance differences among these interpretation methods. We propose an alternative method to create a large interpretation ground truth. Our contributions are summarized as follows:

- We propose IDC, a new benchmark for Interpretation methods for Deep text Classification models. Given a labeled text classification dataset, IDC consists of three methods to create Pseudo interpretation Ground Truth (PGT) datasets. They are gradient-based, mixed integer linear programming, and the hybrid of the two methods. We present the validation of the PGTs with the ground truth of interpretation features 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]. The performance metrics are interpretation recall, interpretation precision, and Cohen's kappa for interpretation features.
- We present the first quantitative evaluation results comparing the effectiveness of these interpretation methods. We will provide the source code to generate PGTs and all the methods we evaluated upon the acceptance of the paper.

The remainder of this paper is organized as follows. In Section 2, we provide background on local interpretation methods. We describe our proposed IDC benchmark in Section 3. We present our evaluation of the six interpretation methods in Section 4. Finally, we give a conclusion in Section 5.

2 Background on Local Interpretation Methods

We divide the existing local interpretation methods into five categories: Probe internal representation, input perturbation, signal backpropagation, gradient-based, and attention-based approaches. Many of these methods were first introduced as interpretation methods for image classification models. Some were later applied for text classification models.

Probe internal representation: This approach examines the input and output signal or the weight of a particular neuron of some layers. 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.

Input perturbation approach: This approach removes part of the input text such as words, phrases, or sentences and calculates the change in the classification probability to measure the importance of the removed part. 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 highly affect the classification confidence.

Signal backpropagation approach: This approach assigns a relevance score to each word in the input by redistribution the output score of the predicted class via backpropagation to each neuron in the previous layer, repeatedly, 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. Different than other interpretation methods, LRP has the ability to 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. Furthermore, there is no parallel GPU implementation publicly available for LRP [14]. Class Activation Map (CAM) [8] interpretation method eliminates the computational overhead by multiply the weight connections of the predicted class with the corresponding feature maps of the last convolutional layer, and then up-sample the generated heatmap of relevance scores. CAM requires a change in the Convolutional Neural Network (CNN) architecture by replacing the fully connected layers by a global average pooling layer.

Gradient-based approach: This approach relies on gradients to lead to the parts of the input 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. The gradient map, called Saliency Map, is generated [10]. Gradient-Class Activation Map (Grad-CAM) [12] multiplies the gradient with respect to the predicted class with the values of the feature maps output of the last convolutional layer (gradient-by-input) and redistributes the multiplication results backward to the input layer via up-sampling. Grad-CAM is applicable to a wide variety of CNN-based models without the need to change 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].

Attention-based approach: Attention-based classification methods have become popular due to their ability to improve the classification accuracy. During training, weights of an attention vector are learned to get the classifier to emphasize more on 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. The learning of the weights of the attention vector(s) adds additional computational complexity during training. Recent works found that using attention weights does not provide an accurate explanation [22, 23]. However, using the learned weights to provide interpretation of a given input is fast.

3 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 important for classification, especially for long documents and datasets with several classes. Both obstacles hinder the creation of a large interpretation ground truth. The existed human annotated datasets like ERASER [20] and E-SNLI [24] provide the annotation only at a clauses or phrases level. Therefore, they cannot be utilized to precisely measure 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, which has a class label per document. We describe the performance metrics on PGTs for 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.

3.1 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. We will explain the WGA function in more details 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 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 over 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. We use M to predict the class label of each training document. Any CNN-based classifiers that offer high classification accuracy can be used. In our implementation, we chose a CNN for text classification [25] with one CNN layer with three window sizes covering 3, 4, and 5 words. The classification performance of CNN models and LSTM models for the text classification task is not much different [16, 26]. Therefore, we chose a simple CNN model to focus on the proposed benchmark itself rather than the classification model. We run M through all the documents in D, and consider only documents that are correctly predicted by M. For each such document, we perform a single backward pass to generate scores for all the words based on their relevance to the correct classification of the document as follows. We adapt the gradient-by-input method for computing relevance scores. The idea of gradient-by-input was introduced and used in several recent gradient-based interpretation methods for image classification such as Grad-CAM and DeepLIFT.

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

Output:
IF_c: Interpretation features Set for class c

Algorithm:
1: $\mathcal{V}$ = ExtractUniqueTerms(X)
2: $RS = \boldsymbol{WGA}(M, D, \mathcal{V})$ #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

Figure 1: Weighted Gram Activation algorithm

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 $f_1, \ldots, f_q$ with a window size of r-gram of words per filter generate (using padding) a corresponding feature map $FM \in \mathbb{R}^{n \times q}$, as shown in Figure 2. Let $GM^c \in \mathbb{R}^{n \times q}$ be a matrix of the gradients with respect to class. We find the maximum gradient for each row in GM^c using max( ), which outputs a gradient vector $\boldsymbol{\delta}^c$ of size $\mathbb{R}^n$ where each element represents the strongest gradient for each r-gram of the words in the document d. Next, we generate the vector $\boldsymbol{z}^c \in \mathbb{R}^n$ with one relevance score for each of the r-grams using Equation (1) to calculate each score value $\boldsymbol{z}^c[j]$ as follows. First, the sum of the product of the highest gradient of the r-gram j and the feature value from the convolution of each filter k to that r-gram is calculated. We apply ReLU( ) to the sum to suppress any negative values. Next, we use $\boldsymbol{z}^c$ to derive relevance scores for the input words for each window size. Note that we use padding to keep the number of the r-grams the same for all window sizes.

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

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

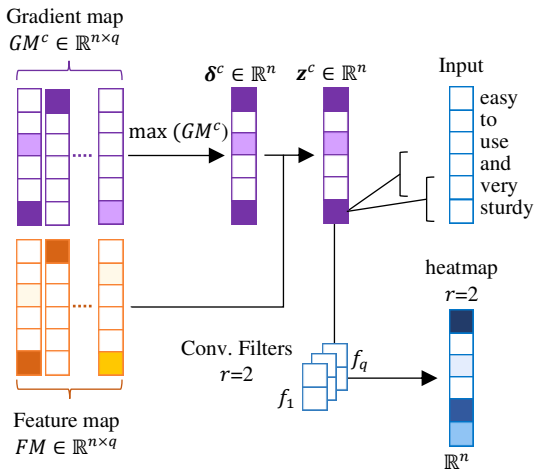

Figure 2: Data flow of WGA( ) function used in Line 2 of Figure 1; n, r, and q represent the number of words, the window size, and the number of filters, respectively.

Gradient-by-input interpretation methods up-sample the feature maps to match the input image dimension. Adapting the idea for text classification is not straightforward. It was found that different words in one r-gram do not necessarily contribute equally in convolution with a particular convolutional filter to generate the feature value [5]. For instance, in Figure 2, the input at position 5 (j =5) which

corresponds to the word "very" is part of the 2-gram "and very". The words "very" and "and" in this gram do not necessarily contribute equally to the feature value at this position $\boldsymbol{z}^c[5]$. Furthermore, each filter of size 2 processes the word "very" twice in the "and very" 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 two-word grams for this filter. We use Equation (2) to distribute the r-gram relevance score of $\boldsymbol{z}^c[j]$ to each word w in the r-gram that includes the word w based on the relevant filter weights of a convolutional filter f.

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

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; and $Conv(\,)$ is the convolutional operation of the r-gram of words that correspond to the convolutional filter f. We calculate the relevance score $s^w(f)$ of the word w for each filter f of the window size r. Thus, for each window size r, we get one heatmap of relevance scores. Finally, we take the average of the relevance scores at the same position across all the heatmaps to get the final heatmap for this document. We find the average of all the relevance scores of the word w of all the documents of class c and store it in the matrix element $RS[w,c] \in [0..1]$. As shown in Figure 2, the more saturated the cell, the more important it is to the classification decision.

3.2 Method 2: Mixed Integer Linear Program (MILP)

The previous method depends on the effectiveness of the chosen text classifier to generate relevance scores for classification. Our second method does not use any classifiers, but analyzes word distribution in each class. First, we construct the set of unique words $\mathcal{V}$ from the training dataset by selecting $|\mathcal{V}|$ words with the highest TF-IDF values. Given $\mathcal{V}$, we want to (1) find the smallest set of interpretation features for each class, (2) restrict the interpretation feature sets of all the classes to be mutually exclusive, and (3) restrict that the chosen interpretation features for a class occur in all the documents of that class and as few as possible in 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 $\boldsymbol{v}^c \in \mathbb{R}^{|\mathcal{V}|}$ be the one hot encoding of IF_c such that when the jth position of $\boldsymbol{v}^c$ is set to 1, the jth word

is selected as an interpretation feature for the class c. The optimization goal is to minimize Equation (3) given the parameter set $\theta = \{\boldsymbol{v}^c, \forall c \in C\}$. Equation (3) is aimed for each interpretation feature set IF_c to have fewest words that appear mostly in the documents of the class c while occurring only 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.

$$\underset{\theta}{\text{minimize}} \ \sum_{c \in C} \sum_{\boldsymbol{d} \in D^c} \sum_{\substack{\hat{c} \in C \\ \hat{c} \neq c}} \boldsymbol{d} \cdot \boldsymbol{v}^{\hat{c}}, \qquad (3)$$

subject to:

$$\sum_{c \in C} \boldsymbol{v}^c[j] \leq 1, \ \forall j = 1..|\mathcal{V}| \qquad (4)$$

$$\boldsymbol{d} \cdot \boldsymbol{v}^c \geq 1, \ \forall c \in C, \ \forall \boldsymbol{d} \in D^c, \qquad (5)$$

where $\boldsymbol{d} \in D^c$ denotes a row of the matrix D^c, which is 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.

3.3 Method 3: Hybrid

We design 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 Figure 1. Then, we construct the set of unique words $\mathcal{V}$ with the words that have the highest relevance score generated by WGA function instead of the TF-IDF representation and solve the mixed-integer linear programing problem as done in Method 2. The goal of using the WGA function is that the words relevance scores have higher correlation to the target class than the TF-IDF values.

3.4 Interpretation Performance Metrics

Given the generated pseudo ground truth datasets, we are able, for the first time in literature, to quantify the effectiveness of existing interpretation methods without relying on classification accuracy. As part of the benchmark, we propose to adapt the widely used metrics: Cohen's kappa, precision, and recall based on interpretation features as the performance metrics. We define the interpretation precision $P_{interp.}$ and interpretation recall $R_{interp.}$ as follows:

$$P_{interp.} = \frac{tp}{tp+fp} \quad (6) \quad \text{and} \quad R_{interp.} = \frac{tp}{tp+fn} \quad (7)$$

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

4 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 interpretation features from the text impact classification confidence? We answer these questions in Sections 4.2, 4.3, and 4.4, respectively.

4.1 Classification Models and Hyperparameters Settings

For all the experiments, we used the CNN classifier for text [25] with one convolutional layer and three window sizes covering 3, 4, and 5 words. We selected the CNN classifier of [25] due to its simplicity and good performance for the text classification task. We used 50 convolutional filters for each window size. We represented each word using a 128-dimension word2vec embedding vector initialized randomly and then optimized during the training. We handled the varying input lengths by padding all the documents to the average length of the documents. We empirically set the training dropout rate to 0.5. We used Adam optimizer with learning rate of 0.001. We used 32 documents per batch. The training process lasted at most 10 epochs for each dataset. We used Python 3 with TensorFlow 2.5 for all our implementation.

For WGA, we set the average relevance threshold α to 0.3 and the difference in the average relevance scores of the two classes β to 0.25 for all the datasets used in our experiments.

4.2 Validation of PGTs with Human Annotators

We validated our PGTs by comparing them with those generated by two human annotators. We randomly sampled without replacement 125 product reviews (rated as 1 star and 2 stars) and 125 product reviews (rated as 4 stars and 5 stars) from the Amazon product reviews [28] test dataset. Since it is rather difficult for the human annotators to provide interpretation ground truth manually for each rating using the

5-star ratings system, two humans individually labeled the same data samples as positive or negative. We discarded the user-reviews with three stars as done in existing works of classification [25]. We provided the two annotators with guidelines on labeling the interpretation features 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 entities. 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.

Compared to the task of labeling rationales, word-level annotations are much more time-consuming and more prone to disagreement. For human-labeling of rationales for text classification tasks, two annotators were used and only 96 text documents were labeled [20]. Labeling of product reviews does not require expertise other than being fluent in English. Both annotators are native English speakers.

To compare the κ inter-agreement of our PGTs and the ground truth by the humans, we generated one set of PGTs for each of the three proposed methods. We generated the PGTs for this experiment from the same Amazon reviews dataset given to the human annotators. The PGTs and the source code of the methods will be made available publicly upon publication of the paper. We combined the generated interpretation features for the 1-star and 2-stars ratings as interpretation features for the negative class and the 4-stars and 5-stars ratings for the positive class. Figure 3 shows the κ inter-agreement value between each PGT and each human annotator.

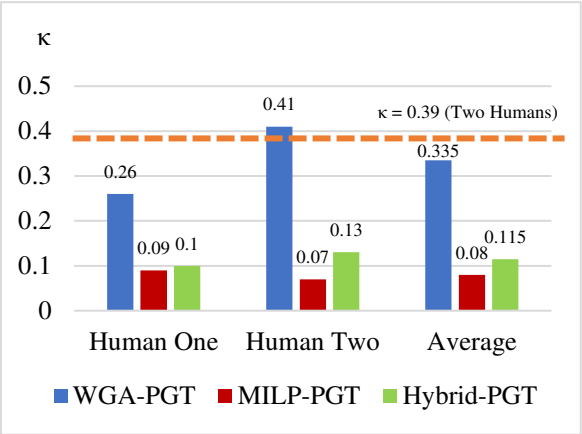

Figure 3: The κ value of 250 user reviews of the Amazon dataset. The κ value between the two human annotators is 0.39 (orange horizontal dash-line).

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. In our experiments, we used Solving Constraint Integer Programs [29]. For a large training dataset like Amazon, we randomly selected a subset from the training dataset of size 5K reviews to construct the PGTs with the MILP and Hybrid methods with a vocabulary size $|\mathcal{V}|$ of 2,000 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 since we provided the two experts 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, which reflects the high confidence of our PGT. We believe that 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.

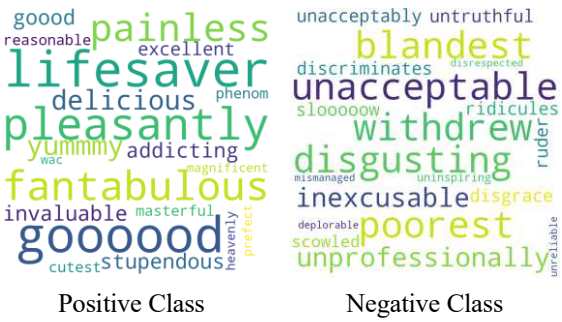

Figure 4: The 20 most relevant interpretation features from WGA-PGT for positive and negative classes of the Amazon reviews training dataset. The higher the WGA score, the larger the word.

4.3 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]. We get the activation scores of the word-level attention as an interpretation map of Hierarchical Attention. We adopted the same implementation of the interpretation methods as referenced in their corresponding papers.

To evaluate the interpretation method against the generated PGTs, we trained the CNN classifier for sentiment analysis (positive and negative) using the Movie Review polarity (MR) [30] dataset. We split the dataset into 60%, 20%, and 20% for training, validation, and testing, respectively. We generated the PGTs by utilizing the training dataset used to train the CNN classifier. We defined the set of the unique words size $|\mathcal{V}|$ to 3,000 words for the MILP and Hybrid methods. To reach the optimal solution for both MILP and Hybrid methods, we discarded the documents that have less than five words from the vocabulary. This reduced the number of training documents used to generate the PGTs to 2K for the MILP method and 7K for the Hybrid method.

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

PGT		Saliency Map	Grad-CAM	Integrated Gradient	DeepLIFT	LRP	Hierarchical Attention
WGA (8K)	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 (2K)	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 (7K)	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

Interpretation Effectiveness

Table 1 shows the interpretation effectiveness using the kappa, interpretation precision, and interpretation recall on the MR dataset. 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 has the potential to perform

better when trained with a full-size dataset and large vocabulary size. The Hybrid method does not perform as well as we expected due to the limitation of mixed integer linear programing. 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 about a margin of error. Hierarchical Attention has the lowest performance. This finding concurs with [22, 23]. Therefore, we conclude that these methods are comparable.

4.4 Impact of Interpretation Features on Classification Confidence

Does removing the PGTs interpretation features from the text impact classification confidence? To answer this question, we measure the average relative classification confidence after removing a subset of the most relevant words to the target class from the input text of the MR test dataset as in Equation (8). The more 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.

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

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.

We measured the drop in relative confidence after removing all and only the interpretation features from the test documents of the MR dataset using the three PGT methods. We also calculated the average number of removed interpretation features from each test document. The three PGT methods removed on average between 15% and 20% of the words from each test document. We define the maximum relative confidence of the CNN text classifier as the relative confidence using the entire text without removing any word. 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 comparing with the maximum. We conclude that these interpretation features greatly impact the text classifier confidence, and thus are important for classification.

5 Conclusion

We present three methods to generate pseudo ground truth of interpretation features for text classification. We found that WGA is good since its average inter-agreement rating is the closest to that of the human annotators. WGA is able to process a large set of documents while MILP is limited by the number of documents and the vocabulary size. Although the designs of WGA and MILP are totally different, both methods agree that the best interpretation methods in terms of interpretation precision and interpretation recall are LRP and the gradient-by-input methods followed by Saliency Map and Hierarchical Attention.

6 Declarations

6.1 Ethics approval and consent to participate

Not applicable.

6.2 Consent for publication

Not applicable.

6.3 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 github.com/xxxxxx

6.4 Competing interests

The authors declare that they have no competing interests.

6.5 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.

6.6 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 reviewed and approved the final manuscript.

6.7 Acknowledgements

Not applicable.

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 (IJCAI). 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 (ICDM). 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 Library Science 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 (ICLR). 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 (WACV). 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 (ICML). 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 (IJCAI). 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 (ICDM). 2017. p. 1033–8.
18. Arras L, Horn F. "What is Relevant in a Text Document ?": An Interpretable Machine Learning Approach. Public Library Science 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. Wiegreffe 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 (EMNLP). 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 Program 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.