Identifying the Source of Vulnerability in Explanation Discrepancy: A Case Study in Neural Text Classification

Ruixuan Tang
University of Virginia
rt5tb@virginia.edu

Hanjie Chen
University of Virginia
hc9mx@virginia.edu

Yangfeng Ji
University of Virginia
yangfeng@virginia.edu

Abstract

Some recent works observed the instability of post-hoc explanations when input side perturbations are applied to the model. This raises the interest and concern in the stability of post-hoc explanations. However, the remaining question is: is the instability caused by the neural network model or the post-hoc explanation method? This work explores the potential source that leads to unstable post-hoc explanations. To separate the influence from the model, we propose a simple *output probability perturbation* method. Compared to prior input side perturbation methods, the *output probability perturbation* method can circumvent the neural model's potential effect on the explanations and allow the analysis on the explanation method. We evaluate the proposed method with three widely-used post-hoc explanation methods (LIME (Ribeiro et al., 2016), Kernel Shapley (Lundberg and Lee, 2017a), and Sample Shapley (Strumbelj and Kononenko, 2010)). The results demonstrate that the post-hoc methods are stable, barely producing discrepant explanations under output probability perturbations. The observation suggests that neural network models may be the primary source of fragile explanations.

1 Introduction

Despite the remarkable performance of neural network models in natural language processing (NLP), the lack of interpretability has raised much concern in terms of their reliability and trustworthiness (Zhang et al., 2021; Doshi-Velez and Kim, 2017; Hooker et al., 2019; Jiang et al., 2018). A common way to improve a model's interpretability is to generate explanations for its predictions from the post-hoc manner. We call these explanations post-hoc explanations (Doshi-Velez and Kim, 2017; Molnar, 2018). Post-hoc explanations demonstrate the relationship between the input text and the model prediction by identifying feature importance scores (Du et al., 2019). In general, a feature with a higher importance score is more important in contributing to the prediction result. Based on feature importance scores, we can select top important features as the model explanation.

However, some recent works (Ghorbani et al., 2019; Subramanya et al., 2019; Zhang et al., 2020; Ivankay et al., 2022; Sinha et al., 2021) have observed explanation discrepancy when input-side perturbation is applied to the model. One question to this observation is what makes the explanation discrepant? Explanations generated by a post-hoc method (Ribeiro et al., 2016; Lundberg and Lee, 2017a; Friedman, 2001) depend on a model's prediction probabilities. If perturbations at the input side cause model prediction probabilities to change, post-hoc explanations may change accordingly.

In Figure 1 (a), we demonstrate a simple example of the process that generates explanations using a post-hoc method. The explanation is generated depending on the probability P. In Figure 1 (b), we demonstrate an example of the same process with perturbation at the input side. The explanation is generated depending on the probability $\bar{P}$. The output probabilities in the two examples are not the same, i.e. $P \neq \bar{P}$. In Figure 1 (a) and (b), it is noticeable that the feature importance score of the same feature has changed. For instance, the feature "love" has different importance scores in the two examples. Since feature importance scores are inconsistent, the explanations in the two examples are different. We call this *explanation discrepancy*, which will be introduced more in subsection 2.2.

However, the prediction label in Figure 1 (a), $\hat{y}$, and the prediction label in Figure 1 (b), $\bar{y}$, are equal, which is $\hat{y} = \bar{y} =$ POSITIVE. This indicates that input side perturbations may not flip the model prediction label, while can make output probabilities change, hence further leading to explanation discrepancy. We argue that, under input side perturbations, it is difficult to identify the source causing the explanation discrepancy. One intuitive justifica-

Proceedings of the Fifth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, pages 356 - 370

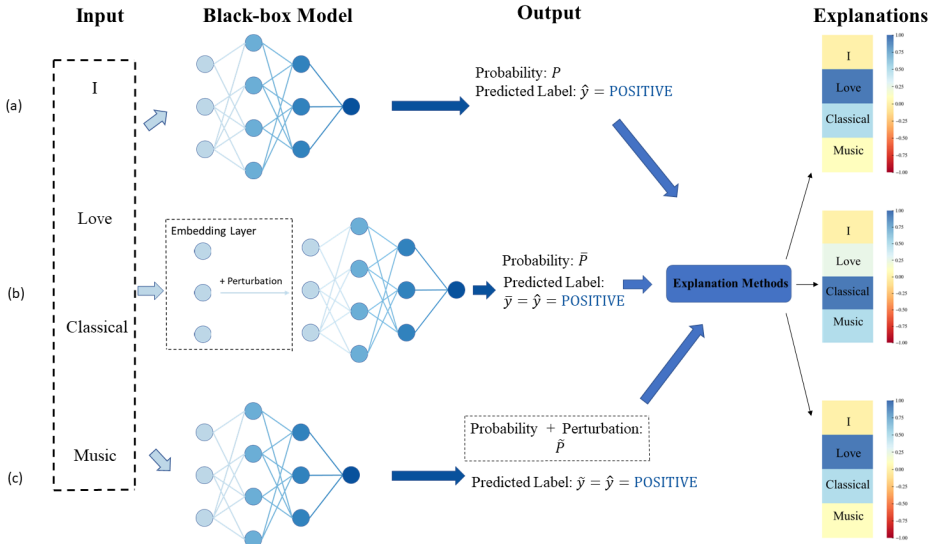

Figure 1: The pipeline of a simple example that post-hoc explanation methods generate explanations with (a) no perturbation applied. (b) perturbation applied at the input side. (c) perturbation applied at the output probabilities.

tion is that the perturbation at the input side has to pass through both the model and the post-hoc explanation method. Both the model and the post-hoc explanation method are possible factors that result in unstable explanations. For example, the model's prediction behavior may change under input side perturbations, that is focusing on different features to make predictions, hence resulting in the explanation discrepancy (Chen and Ji, 2020, 2022). Or the explanation method itself may be vulnerable to input perturbations, producing discrepant explanations. The instability may not be told from the prediction results, but reflected in the explanations, i.e., explanation discrepancy

In this paper, we propose a simple strategy to demonstrate the potential source that causes explanation discrepancy. To circumvent the potential influence of the model on the explanations, we design an *output probability perturbation* method by slightly modifying the prediction probabilities, as shown in Figure 1 (c). In this work, we focus on the model-agnostic post-hoc methods, LIME (Ribeiro et al., 2016), Kernel Shapley (Lundberg and Lee, 2017a), and Sample Shapley (Strumbelj and Kononenko, 2010), that explain the black-box models. If a similar explanation discrepancy can be observed when only output probability perturbation is applied, it would suggest that post-hoc explanation methods may be unstable because the potential influence from the black-box model has been blocked. Otherwise, we should not blame post-hoc explanation methods as the source of vulnerability in fragile explanations (Sinha et al., 2021; Subramanya et al., 2019).

2 Method

2.1 Background

For a text classification task, $\boldsymbol{x}$ denotes the input text consisting of N words, $\boldsymbol{x} = [\boldsymbol{x}^{(1)}, \cdots, \boldsymbol{x}^{(N)}]$, with each component $\boldsymbol{x}^{(n)} \in R^d$ representing the n-th word embedding. We define a black-box classifier as $f(\cdot)$ and its output probability of a given $\boldsymbol{x}$ on the corresponding label k is $P(y = k \mid \boldsymbol{x}) = f_k(\boldsymbol{x})$, where $k \in \{1, \ldots, C\}$ and C is the total number of label classes.

To explain a black-box model's prediction $\hat{y} = f(\boldsymbol{x})$, a class of post-hoc explanation methods approximate the model locally via additive feature attributions (Lundberg and Lee, 2017b; Ribeiro et al., 2016; Shrikumar et al., 2017). Specifically, these algorithms demonstrate the relationship between the input text and the prediction result by evaluating the contribution of each input feature to the model prediction result. These methods would assign a feature importance score to each input feature to represent its contribution to the prediction. We use LIME (Ribeiro et al., 2016) as an example.

Example: Post-hoc Explanation Method, LIME. It first sub-samples words from the input, $\boldsymbol{x}$, to form a list of pseudo examples $\{\boldsymbol{z}_{j=1}^{L}\}$, and then the contributions of input features are estimated by a linear approximation $f_{\hat{y}}(\boldsymbol{r}) \approx g_{\hat{y}}(\boldsymbol{r'})$, where $\boldsymbol{r} \in \{\boldsymbol{x}, \boldsymbol{z}_{j=1}^{L}\}$, $g_{\hat{y}}(\boldsymbol{r}) = \boldsymbol{w}_{\hat{y}}^{T}\boldsymbol{r'}$, and $\boldsymbol{r'}$ is a simple

representation of $\boldsymbol{r}$, e.g. bag-of-words representation. The weights $\{w_{\hat{y}}^{(n)}\}$ represent importance scores of input features $\{\boldsymbol{x}^{(n)}\}$. Let $I(\boldsymbol{x}, \hat{y}, P)$ denote the explanation for the model prediction on $\boldsymbol{x}$, where $\hat{y}$ is the predicted label and P represents output probabilities.

2.2 Explanation Discrepancy

As mentioned in the previous section, the explanation discrepancy may happen when input perturbations are applied to the model. Let $I(\boldsymbol{x}, \hat{y}, P)$ and $I(\bar{\boldsymbol{x}}, \bar{y}, \bar{P})$ denote the explanation to the model prediction based on the original input $\boldsymbol{x}$ and the perturbed input $\bar{\boldsymbol{x}}$ respectively, where $\bar{\boldsymbol{x}} = \boldsymbol{x} + \varepsilon$, and ε is the perturbation at input. Similarly, we define $I(\boldsymbol{x}, \widetilde{y}, \widetilde{P})$ as the explanation to the prediction based on the perturbed output probability $\widetilde{P} = P + \varepsilon'$, where ε' is the perturbation on the output probability. Note that when ε and ε' are small, the model prediction stay the same, which is $\hat{y} = \bar{y} = \widetilde{y}$. The explanation discrepancy between $I(\bar{\boldsymbol{x}}, \bar{y}, \bar{P})$ and $I(\boldsymbol{x}, \hat{y}, P)$ is denoted as δ_{input}, and the discrepancy between $I(\boldsymbol{x}, \widetilde{y}, \widetilde{P})$ and $I(\boldsymbol{x}, \hat{y}, P)$ is denoted as δ_{output}.

We use Figure 1 in section 1 as an example to illustrate explanation discrepancy in details. The explanation, $I(\boldsymbol{x}, \hat{y}, P)$, in Figure 1 (a) is "Love", "Classical", "I" and "Music", in the descending order of importance scores. The explanation, $I(\bar{\boldsymbol{x}}, \bar{y}, \bar{P})$, in Figure 1 (b) is "Classical", "Music", "Love", and "I", in the descending order. The explanation, $I(\boldsymbol{x}, \hat{y}, \widetilde{P})$, in Figure 1 (c) is "Love", "Classical", "I" and "Music", in the descending order. Generally, after perturbation, explanation inconsistency reflects in two aspects. The first aspect is whether the overall ranking of the features based on their importance scores in the explanation remains the same. For example, "Love" ranks the first in the explanation in Figure 1 (a), while drops to the third in the explanation in Figure 1 (b). The discrepancy is denoted as δ_{input}. The second aspect is whether the top K important features in the explanation are consistent. For example, if $K = 2$, the first two important words in Figure 1 (a) are "Love" and "Classical", while those in Figure 1 (b) are "Classical" and "Music". The difference can also be denoted as δ_{input} mentioned above. Similarly, the same aspect of explanation discrepancy in Figure 1 (a) and Figure 1 (c) can be denoted as δ_{output}.

2.3 Output Probability Perturbation Method

As mentioned in section 1, the limitation of input perturbation methods is the difficulty in identifying the primary source that causes explanation discrepancy. Motivated by this, we propose the output probability perturbation method to circumvent the influence of black-box models.

Specifically, given an example $\boldsymbol{x}$, we add a small perturbation to the model output probabilities $\{P(y = k \mid \boldsymbol{x}) + \varepsilon'_{y=k}\}_{k=1}^{C}$. To guarantee the modified $\{P(y = k \mid \boldsymbol{x}) + \varepsilon'_{y=k}\}_{k=1}^{C}$ are still legitimate probabilities, we further normalize them as

$$\widetilde{P}(y = k \mid \boldsymbol{x}) = \frac{P(y = k \mid \boldsymbol{x}) + \varepsilon'_{y=k}}{\sum_{i=1}^{C}\{P(y = i \mid \boldsymbol{x}) + \varepsilon'_{y=i}\}} \tag{1}$$

The explanation in the case with output probability perturbation is computed based on the output probability $\widetilde{P}(y = \hat{y} \mid \boldsymbol{x})$. The proposed method well suits the motivation of investigating the source that causes explanation discrepancy. The main reason is that, unlike perturbation applied at the input side, the proposed method avoids the potential effects of the model's vulnerability on post-hoc explanations. We use LIME (Ribeiro et al., 2016) as an example to demonstrate the proposed method.

Example: Output probability perturbation in LIME algorithm. As denoted in subsection 2.1, $\boldsymbol{r'}$ is the bag-of-words representation of the original input text, $\boldsymbol{x}$. A simplified version [1] of LIME algorithm is equivalent to finding a solution of the following linear equation:

$$\boldsymbol{w}_{\hat{y}}^{T}\boldsymbol{r'} = \widetilde{\boldsymbol{p}}_{\hat{y}} \tag{2}$$

where $\widetilde{\boldsymbol{p}}_{\hat{y}} = [\widetilde{P}(y = \hat{y} \mid \boldsymbol{x}), \widetilde{P}(y = \hat{y} \mid \boldsymbol{z}_1), \ldots, \widetilde{P}(y = \hat{y} \mid \boldsymbol{z}_L)]^T$ are the perturbed probabilities on the label $\hat{y}$, and $\boldsymbol{w}_{\hat{y}}^{T}$ is the weight vector, where each element measures the contribution of an input word to the prediction $\hat{y}$. A typical explanation from LIME consists of top important words according to $\boldsymbol{w}_{\hat{y}}$. Essentially, the proposed output perturbation is similar to the perturbation analysis in linear systems (Golub and Van Loan, 2013), which aims to identify the stability of these systems. Despite the simple formulation in Equation 2, a similar linear system can also be used to explain the Shapley-based explanation methods

[1] Without the example weight computed from a kernel function and the regularization term of explanation complexity.

(e.g., Sample Shapley (Strumbelj and Kononenko, 2010)).

3 Experiment

3.1 Experiment Setup

Datasets. We adopt four text classification datasets: IMDB movie reviews dataset (Maas et al., 2011, IMDB), AG's news dataset (Zhang et al., 2015, AG's News), Stanford Sentiment Treebank dataset with binary labels (Socher et al., 2013, SST-2), and 6-class questions classification dataset TREC (Li and Roth, 2002, TREC). The summary statistics of datasets are shown in Table 1.

Models. We apply three neural network models, Convolutional Neural Network (Kim, 2014, CNN), Long Short Term Memory Network (Hochreiter and Schmidhuber, 1997, LSTM), and Bidirectional Encoder Representations from Transformers (Devlin et al., 2018, BERT).

The principle of CNN model is based on information processing in the visual system of humans. The core characteristics are that it can efficiently decrease the dimension of input, and it can efficiently retain important features of the input (Kim, 2014).

LSTM model is one advanced RNN model. Unlike the architecture of a standard feedforward deep learning neural network, it has feedback connections in the architecture, which helps to process sequential data (e.g., language and speech) (Hochreiter and Schmidhuber, 1997, LSTM).

BERT model is a Language Model (LM). In the NLP research, the main tasks of the BERT model are (1) Sentence pairs classification tasks and (2) Single sentence classification tasks (Devlin et al., 2018). In this work, we focus on the second task while we apply the BERT model in the experiment.

The prediction performance of the three models on the four datasets are recorded in Table 2.

Post-hoc Explanation Methods. We adopt three post-hoc explanation methods, Local Interpretable Model-Agnostic Explanations (Ribeiro et al., 2016, LIME), Kernel Shapley (Lundberg and Lee, 2017a), and Sample Shapley (Strumbelj and Kononenko, 2010). LIME, Kernel Shapley, and Sample Shapley are additive feature attribution methods. The additive feature method provides a feature importance score on every feature for each text input based on the model prediction.

LIME and Kernel Shapley are two post-hoc methods adopting a similar strategy. The first step is to generate a set of pseudo examples and their corresponding labels based on the black-box model's predictions on them (Ribeiro et al., 2016; Lundberg and Lee, 2017a). The second step is to train an explainable machine learning model (eg: linear regression, LASSO) with the pseudo examples (Ribeiro et al., 2016; Lundberg and Lee, 2017a). The difference between the LIME algorithm and the Kernel Shapley algorithm is in the way to calculate the weight of pseudo examples in the explainable model (Molnar, 2018). LIME algorithm relies on the distance between the original example and the pseudo example (Ribeiro et al., 2016). Kernel Shapley algorithm relies on the Shapley value estimation (Lundberg and Lee, 2017a).

Sample Shapley is a post-hoc method based on Shapley value (Shapley, 1953a), which stems from coalitional game theory. Shapley value provides an axiomatic solution to attribute the contribution of each word in a fair way. However, the exponential complexity of computing Shapley value is intractable. Sampling Shapley (Strumbelj and Kononenko, 2010) provides a solvable approximation to Shapley value via sampling.

Evaluation Metrics. In the experiment, we apply two evaluation metrics, Kendall's Tau order rank correlation score, and the Top-K important words overlap score (Chen et al., 2019; Kendall, 1938; Ghorbani et al., 2019) to evaluate the discrepancy between explanations (i.e., δ_{input} and δ_{output}).

As illustrated in subsection 2.2, explanation discrepancy can be evaluated in in two aspects. We use Kendall's Tau order rank correlation score to quantify the change of the overall ranking of feature importance scores in explanations. For example, in Figure 1 (a) and (b), we can apply Kendall's Tau order rank correlation score to identify how close the overall ranking of features in the two examples. If the score is close to 1, then the two explanations are similar. If the score is close to -1, then the two explanations differ significantly. We use Top-K important words overlap score to evaluate the discrepancy on the top K features in the explanations. This metric computes the overlap ratio among the top K features. In this work, we set $K = 5$.

Dataset	C	L	$\#train$	$\#dev$	$\#test$	vocab	threshold	length
IMDB	2	268	20K	5K	25K	29571	5	250
SST-2	2	19	6920	872	1821	16190	0	50
AG's News	4	32	114K	6K	7.6K	21838	5	50
TREC	6	10	5000	452	500	8026	0	15

Table 1: Summary statistics for the datasets where C is the number of classes, L is the average sentence length, # counts the number of examples in train/dev/test sets, vocab is the vocab size, and the threshold is the low-frequency threshold, and length is mini-batch sentence length.

Dataset	**CNN**	**LSTM**	**BERT**
IMDB	86.30	86.60	90.80
SST-2	82.48	80.83	91.82
AG's News	89.90	88.90	95.10
TREC	92.41	90.80	97.00

Table 2: Prediction accuracy(%) of the three neural network models (CNN, LSTM and BERT) on the four datasets (IMDB, SST-2, AG's News and TREC).

3.2 Explanation Discrepancy Comparison Experiment

To explore the primary source causing fragile explanations, we conduct a comparison experiment to evaluate and compare between explanation discrepancy δ_{input}, and explanation discrepancy δ_{output}. The definition of δ_{input}, and δ_{output} are introduced in subsection 2.2. δ_{input} denotes the discrepancy between the explanation generated by the black-box model with no perturbation, $I(\boldsymbol{x}, \hat{y}, P)$, and the explanation generated by the black-box model with perturbation at the input, $I(\bar{\boldsymbol{x}}, \hat{y}, \bar{P})$. While δ_{output} denotes the discrepancy of $I(\boldsymbol{x}, \hat{y}, P)$ and the explanation generated by the black-box model with perturbation at the output probability, $I(\bar{\boldsymbol{x}}, \hat{y}, \widetilde{P})$.

In this experiment, for output probability perturbation, we directly add random noise to the model output probabilities. For comparison, we add the noise to word embeddings for input perturbations (Liu et al., 2020). Both input side perturbation and output probability perturbation are applied with noise sampled from a Gaussian distribution, $\mathcal{N}(0, \sigma^2)$. We apply Gaussian noise because it is easy to control the perturbation level by modifying the variance of the Gaussian distribution σ^2. In experiments, we applied five different perturbation levels from "0" to "4". "0" means the slightest perturbation level, zero perturbation, while "4" represents the strongest perturbation level. The specific value of each perturbation level is shown in Table 3.

Note that for each level, the input side perturbations and the output probability perturbations are different because we select different perturbations for the input side and the output probability to reach a similar accuracy at each level. If the model's accuracy is not close at each level, it is difficult to evaluate the results.

Perturbation Source	**Level**	σ^2
Input Side (σ^2_{input})	0	0
	1	0.05
	2	0.1
	3	0.15
	4	0.2
Output Probability (σ^2_{output})	0	0
	1	0.25
	2	0.5
	3	0.75
	4	1

Table 3: Perturbation levels applied to the input and output respectively.

3.3 Results and Discussion

Figure 2 shows the results of the IMDB dataset. Due to the page limit, full results of other datasets are shown in Figure 4, Figure 5 and Figure 6 in Appendix A, which have similar tendencies. Kendall's Tau order rank correlation score plots are shown in Figure 2 (a), (b) and (c). Top-K important words overlap score plots are shown in Figure 2 (d), (e) and (f). Figure 2 (a) and (d) show the results of the LIME method. Figure 2 (b) and (e) show the results of the Kernel Shapley method. Figure 2 (c) and (f) show the results of the Sample Shapley method.

Kendall's Tau order rank correlation score evaluation results. Kendall's Tau order rank correlation score results indirectly illustrate the stability of post-hoc explanation methods. Furthermore, previous observation on the explanation difference can

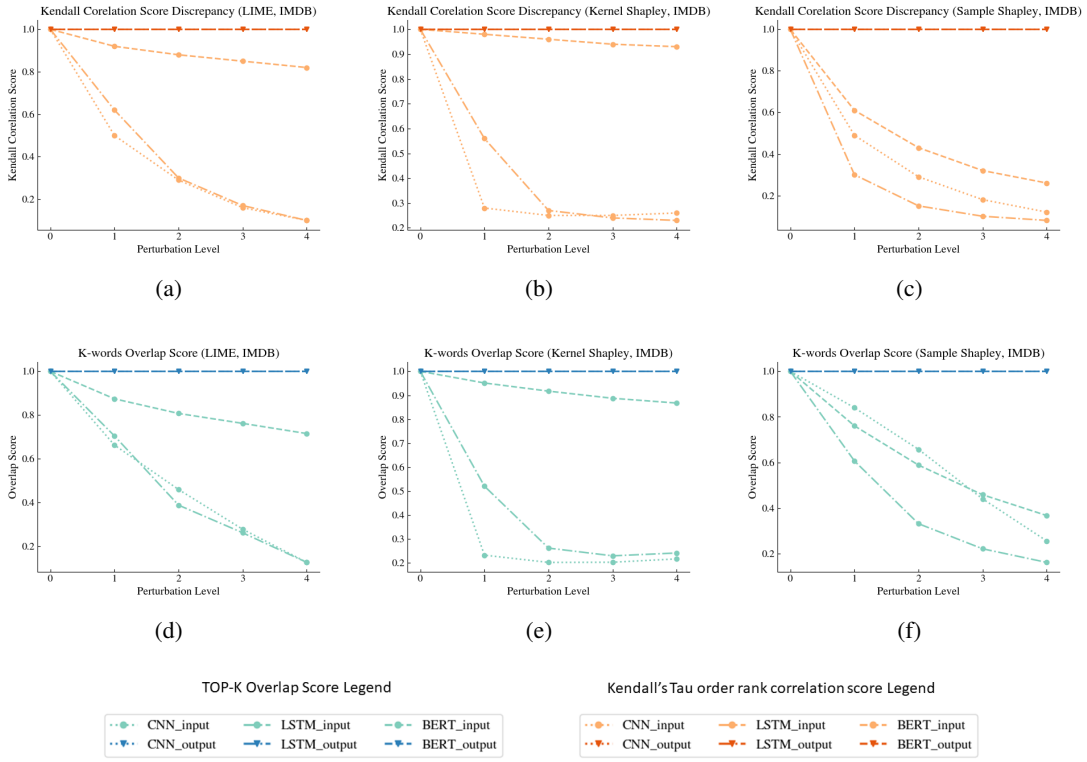

Figure 2: Comparison experiment results on the IMDB dataset; (a) and (d) demonstrate results using LIME method; (b) and (e) demonstrate results using Kernel Sharply method; (c) and (f) demonstrate results using Sample Sharply method.

be attributed to the potential influence caused by the black-box model. In Figure 2 (a), (b) and (c), the large gap between δ_{input} and δ_{output} is consistent across all three post-hoc explanation methods, LIME, Kernel Shapley and Sample Shapley. For output probability perturbations, it is noticeable that the values of Kendall's Tau order rank correlation scores remain the same with the perturbation level increasing from "0" to "4". This indicates that the overall ranking of feature importance scores are stable under output perturbations. Furthermore, the results suggest that for a given input, if $\boldsymbol{x}$ and prediction results stay unchanged, $\hat{y} = \widetilde{y}$, the only perturbation ε' at output probability is unlikely to influence explanations generated by the post-hoc methods. In other words, the explanation discrepancy observed in the previous study (Ivankay et al., 2022; Sinha et al., 2021) is unlikely caused by the post-hoc methods. Meanwhile, for the baseline results (perturbation applied at the input), it is notifiable that the values of Kendall's Tau order rank correlation scores decrease obviously with the increase of input perturbation intensity levels. This indicates that the black-box model is vulnerable to input perturbations, which causes fragile explanations. Based on the observations, we claim that the black-box model is more likely to be the primary source that results in fragile explanations.

Top-K word importance score evaluation results. Top-K word importance score evaluation shows the same result: the black-box model is the primary source causing explanation discrepancy. In Figure 2 (d), (e) and (f), δ_{input} against δ_{output} display an obvious discrepancy across the three post-hoc explanation methods. For output probability perturbations, δ_{output} shows no change in the overlap among the top K important words. This indicates that, for the top five important features in the explanation of each corresponding prediction result, output probability perturbations will not cause any difference. The results under this metric also illustrate that the black-box model is more likely to cause fragile explanations than explanation methods themselves.

3.4 Further Analysis on LIME Algorithm

According to the previous results, we have a conclusion that post-hoc explanation methods are stable.

We further analyze the stability of the explanation algorithms. We use the LIME algorithm (Ribeiro et al., 2016) as an example.

$$L(f_{\hat{y}}(\boldsymbol{r}), g_{\hat{y}}(\boldsymbol{r'}), \pi) = \sum_{\boldsymbol{r},\boldsymbol{r'} \in \boldsymbol{R}} \pi(f_{\hat{y}}(\boldsymbol{r}) - g_{\hat{y}}(\boldsymbol{r'})) \tag{3}$$

Equation 3 is definition of the loss function in LIME algorithm (Ribeiro et al., 2016). In the loss function, $\pi g_{\hat{y}}(\boldsymbol{r'})$ is denoted the kernel calculation function of the algorithm. $\boldsymbol{r'}$ represents the pseudo example based on the original example, $\boldsymbol{r}$. $g_{\hat{y}}(\boldsymbol{r'})$ represents the linear local explainable model on the pseudo example, $\boldsymbol{r'}$. Here, we use a general linear model representation to represent the explainable model, $g_{\hat{y}}(\boldsymbol{r}) = \boldsymbol{w}_{\hat{y}}^T \boldsymbol{r'}$. $\boldsymbol{w}_{\hat{y}}^T$ is the weight function of the linear model and it is the importance feature score calculation function as well. Equation 4 is the kernel calculation function of the LIME algorithm after expanding.

$$G = \pi g_{\hat{y}}(\boldsymbol{r}) = \pi \boldsymbol{w}_{\hat{y}}^T \boldsymbol{r'} \tag{4}$$

The form of the kernel calculation function can be interpreted as a general linear function, $Ax = b$. In the linear function, the condition number, (κ), is applied to evaluate how sensitive the linear function is due to a small change at the input and reflects in its output (Belsley et al., 2005). If the condition number, (κ), which is the largest eigenvalue in the matrix A divided by the smallest eigenvalue in the matrix A, is large, the solution x would change rapidly by a slight difference in b, which would cause sensitivity of the solution to the slight error in the input (Goodfellow et al., 2016). In Equation 4, $\pi \boldsymbol{r'}$ can be considered as the matrix A, and the feature importance score function $\boldsymbol{w}_{\hat{y}}^T$ can be considered as the solution x. If $\pi \boldsymbol{r'}$ is a stable linear system, the feature importance score function $\boldsymbol{w}_{\hat{y}}^T$ would be unlikely sensitive to a minor change at the linear system input side, and the corresponding post-hoc explanation method is stable. The form of the kernel calculation function can be interpreted as a general linear function, $Ax = b$. In the linear function, the condition number, (κ), is applied to evaluate how sensitive the linear function is due to a small change at the input and reflects in its output (Belsley et al., 2005). If the condition number, (κ), which is the largest eigenvalue in the matrix A divided by the smallest eigenvalue in the matrix A, is large, the solution x would change rapidly by a slight difference in b, which would cause sensitivity of the solution to the slight error in the input (Goodfellow et al., 2016). In Equation 4, $\pi \boldsymbol{r'}$ can be considered as the matrix A, and the feature importance score function $vecw_{\hat{y}}^T$ can be considered as the solution $\boldsymbol{r}$. If $\pi \boldsymbol{r'}$ is a stable linear system, the feature importance score function $\boldsymbol{w}_{\hat{y}}^T$ would be unlikely sensitive to a minor change at the linear system input side, and the corresponding post-hoc explanation method is stable. Since the kernel function is a pure numerical step without semantics involved. We conduct a simulation experiment to explore the stability of the LIME algorithm (Ribeiro et al., 2016).

Simulation Experiment and Results In the simulation experiment, the pseudo example, $\boldsymbol{r'}$, is a matrix with the size of sub-sampling size, m, multiple with the length of a sentence, l. We select $m = 200$, which is the actual sample size value we applied in the comparison experiment. For the sentence length, first, we simulate the case when sentence length is fixed, that is $l = 20$. Then, to compare condition number distribution when sentence length is different, we apply two more cases, that are $l = 30$, and $l = 40$. For each length, we simulate it for 500 iterations. For π, it is the distance between the original input to the sub-sampling based on the original input in the LIME algorithm (Ribeiro et al., 2016). In the simulation experiment, we apply cosine distance to represent the value of π.

Total iteration number	$\kappa \in [5, 6)$	$\kappa \in [6, 7)$
500	392	108

Table 4: Condition number (κ) distribution when $l = 20$

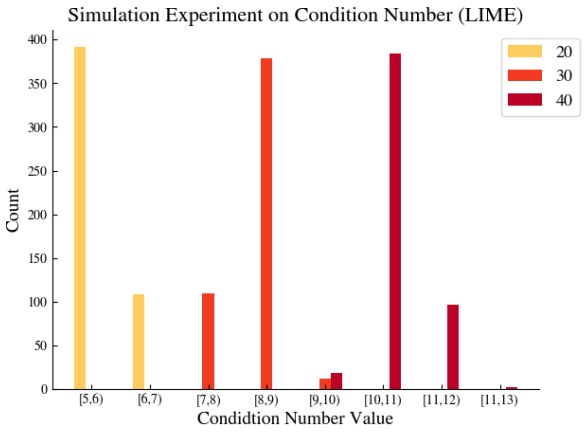

Figure 3: Simulation Experiment Result.

In Table 4, the result of the simulation experiment when the sentence length is fixed, $l = 20$, demonstrates that the majority of the condition number κ of the matrix $\pi \boldsymbol{r'}$ is lower than 7. In Goldstein's work, it suggests that the condition number of a stable or a well-conditioning matrix should be lower than 30 (Goldstein, 1993). It means that the feature importance score function, W, is less likely influenced by a small perturbation involved, which is also reflected in the real dataset results in the comparison experiment in subsection 3.3. In Figure 3, the result of the simulation experiment when sentence lengths are different shows that when the length of the sentence increases, the condition number κ of the matrix $\pi \boldsymbol{r'}$ increases with a tiny amplitude. The majority of the condition number κ of the matrix is lower than 13 when the length is from 20 to 40. Although the result demonstrates that the condition number κ would increase with sentence length increasing, the increasing amplitude is small and the majority of the condition number is lower than the threshold number. The result suggests that the matrix $\pi \boldsymbol{r'}$ in the LIME algorithm can remain a small condition number, which makes the linear system relatively stable. In other words, the LIME algorithm (Ribeiro et al., 2016) is a relatively stable post-hoc explanation method.

4 Previous Works

Post-hoc Explanation Methods Most works focus on explaining neural network models in a post-hoc manner, especially generating a local explanation for each model prediction. The white-box explanation methods, such as gradient-based explanations (Hechtlinger, 2016), and attention-based explanations (Ghaeini et al., 2018), either require additional information (e.g. gradients) from the model or incur much debates regarding their faithfulness to model predictions (Jain and Wallace, 2019).

Another line of work focuses on explaining black-box models in the model-agnostic way. Li et al. (2016) proposed a perturbation-based explanation method, Leave-one-out, that attributes feature importance to model predictions by erasing input features one by one. Ribeiro et al. (2016) proposed to estimate feature contributions locally via linear approximation based on pseudo examples. Some other works proposed the variants of the Shapley value (Shapley, 1953b) to measure feature importance, such as the Sample Shapley method (Strumbelj and Kononenko, 2010), the Kernel Shapley method (Lundberg and Lee, 2017a), and the L/C-Shapley method(Chen et al., 2018).

Model robustness Recent works have shown the vulnerability of model due to adversarial attacks (Szegedy et al., 2013; Goodfellow et al., 2014; Zhao et al., 2017). Some adversarial examples are similar to original examples but can quickly flip model predictions, which causes concern on model robustness (Jia et al., 2019). In the text domain, a common way to generate adversarial examples is by heuristically manipulating the input text, such as replacing words with their synonyms (Alzantot et al., 2018; Ren et al., 2019; Jin et al., 2020), misspelling words (Li et al., 2018; Gao et al., 2018), inserting/removing words (Liang et al., 2017), or concatenating triggers (Wallace et al., 2019).

Explanation Robustness Previous work explored explanation robustness by either perturbing the inputs (Ghorbani et al., 2019; Subramanya et al., 2019; Zhang et al., 2020; Heo et al., 2019) or manipulating the model (Wang et al., 2020; Slack et al., 2020; Zafar et al., 2021). For example, Slack's group fooled post-hoc explanation methods by hiding the bias for black-box models based on the proposed novel scaffolding technique (Slack et al., 2020). However, all of these works cannot disentangle the sources that cause fragile explanation. Differently, the proposed method mitigates the influence of model to the explanation by perturbing model outputs.

5 Conclusion

In this work, our main contribution is to identify the primary source of fragile explanation, where we propose an output probability perturbation method. With the help of this proposed method, observation results can illustrate a conclusion that the primary potential source that caused fragile explanations in the previous studies is the black-box model itself, which also illustrate that some limitations of prior methods. Furthermore, in subsection 3.4, we analyze the kernel calculation inside the LIME algorithm (Ribeiro et al., 2016). We apply the condition number of the matrix and simulation experiments to demonstrate that the kernel calculation matrix inside LIME has a low condition number. This result further suggests the stability of the LIME algorithm.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. *arXiv preprint arXiv:1804.07998*.

David A Belsley, Edwin Kuh, and Roy E Welsch. 2005. *Regression diagnostics: Identifying influential data and sources of collinearity*. John Wiley & Sons.

Hanjie Chen and Yangfeng Ji. 2020. Learning variational word masks to improve the interpretability of neural text classifiers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4236–4251, Online. Association for Computational Linguistics.

Hanjie Chen and Yangfeng Ji. 2022. Adversarial training for improving model robustness? look at both prediction and interpretation. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Jianbo Chen, Le Song, Martin J Wainwright, and Michael I Jordan. 2018. L-shapley and c-shapley: Efficient model interpretation for structured data. *arXiv preprint arXiv:1808.02610*.

Jiefeng Chen, Xi Wu, Vaibhav Rastogi, Yingyu Liang, and Somesh Jha. 2019. Robust attribution regularization. *arXiv preprint arXiv:1905.09957*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.

Mengnan Du, Ninghao Liu, and Xia Hu. 2019. Techniques for interpretable machine learning. *Communications of the ACM*, 63(1):68–77.

Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56. IEEE.

Reza Ghaeini, Xiaoli Z Fern, and Prasad Tadepalli. 2018. Interpreting recurrent and attention-based neural models: a case study on natural language inference. *arXiv preprint arXiv:1808.03894*.

Amirata Ghorbani, Abubakar Abid, and James Zou. 2019. Interpretation of neural networks is fragile. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3681–3688.

Richard Goldstein. 1993. Conditioning diagnostics: Collinearity and weak data in regression.

Gene H Golub and Charles F Van Loan. 2013. *Matrix computations*, volume 3. JHU press.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Yotam Hechtlinger. 2016. Interpretation of prediction models using the input gradient. *arXiv preprint arXiv:1611.07634*.

Juyeon Heo, Sunghwan Joo, and Taesup Moon. 2019. Fooling neural network interpretations via adversarial model manipulation. *arXiv preprint arXiv:1902.02041*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. 2019. A benchmark for interpretability methods in deep neural networks. *Advances in neural information processing systems*, 32.

Adam Ivankay, Ivan Girardi, Chiara Marchiori, and Pascal Frossard. 2022. Fooling explanations in text classifiers. *arXiv preprint arXiv:2206.03178*.

Sarthak Jain and Byron C Wallace. 2019. Attention is not explanation. *arXiv preprint arXiv:1902.10186*.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. *arXiv preprint arXiv:1909.00986*.

Heinrich Jiang, Been Kim, Melody Guan, and Maya Gupta. 2018. To trust or not to trust a classifier. *Advances in neural information processing systems*, 31.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8018–8025.

Maurice G Kendall. 1938. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2018. Textbugger: Generating adversarial text against real-world applications. *arXiv preprint arXiv:1812.05271*.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. 2017. Deep text classification can be fooled. *arXiv preprint arXiv:1704.08006*.

Xiaodong Liu, Hao Cheng, Pengcheng He, Weizhu Chen, Yu Wang, Hoifung Poon, and Jianfeng Gao. 2020. Adversarial training for large neural language models. *arXiv preprint arXiv:2004.08994*.

Scott M Lundberg and Su-In Lee. 2017a. A unified approach to interpreting model predictions. In *Proceedings of the 31st international conference on neural information processing systems*, pages 4768–4777.

Scott M Lundberg and Su-In Lee. 2017b. A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150.

Christoph Molnar. 2018. A guide for making black box models explainable. *URL: https://christophm. github. io/interpretable-ml-book*.

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 1085–1097.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "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, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144.

Lloyd S Shapley. 1953a. A value for n-person games. *Contributions to the Theory of Games*, 2(28).

LS Shapley. 1953b. Quota solutions op n-person games1. *Edited by Emil Artin and Marston Morse*, page 343.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning important features through propagating activation differences. In *International Conference on Machine Learning*, pages 3145–3153. PMLR.

Sanchit Sinha, Hanjie Chen, Arshdeep Sekhon, Yangfeng Ji, and Yanjun Qi. 2021. Perturbing inputs for fragile interpretations in deep natural language processing. *arXiv preprint arXiv:2108.04990*.

Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. 2020. Fooling lime and shap: Adversarial attacks on post hoc explanation methods. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 180–186.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Erik Strumbelj and Igor Kononenko. 2010. An efficient explanation of individual classifications using game theory. *The Journal of Machine Learning Research*, 11:1–18.

Akshayvarun Subramanya, Vipin Pillai, and Hamed Pirsiavash. 2019. Fooling network interpretation in image classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2020–2029.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing nlp. *arXiv preprint arXiv:1908.07125*.

Junlin Wang, Jens Tuyls, Eric Wallace, and Sameer Singh. 2020. Gradient-based analysis of nlp models is manipulable. *arXiv preprint arXiv:2010.05419*.

Muhammad Bilal Zafar, Michele Donini, Dylan Slack, Cédric Archambeau, Sanjiv Das, and Krishnaram Kenthapadi. 2021. On the lack of robust interpretability of neural text classifiers. *arXiv preprint arXiv:2106.04631*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *arXiv preprint arXiv:1509.01626*.

Xinyang Zhang, Ningfei Wang, Hua Shen, Shouling Ji, Xiapu Luo, and Ting Wang. 2020. Interpretable deep learning under fire. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*.

Yu Zhang, Peter Tiňo, Aleš Leonardis, and Ke Tang. 2021. A survey on neural network interpretability. *IEEE Transactions on Emerging Topics in Computational Intelligence*.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. 2017. Generating natural adversarial examples. *arXiv preprint arXiv:1710.11342*.

A Figures of Comparison Experiments Result on SST-2, AG's News and TREC Dataset

In the section, we include comparison experiments results of the SST-2 dataset in Figure 4, the AG's News dataset in Figure 5, and the TREC dataset in Figure 6.

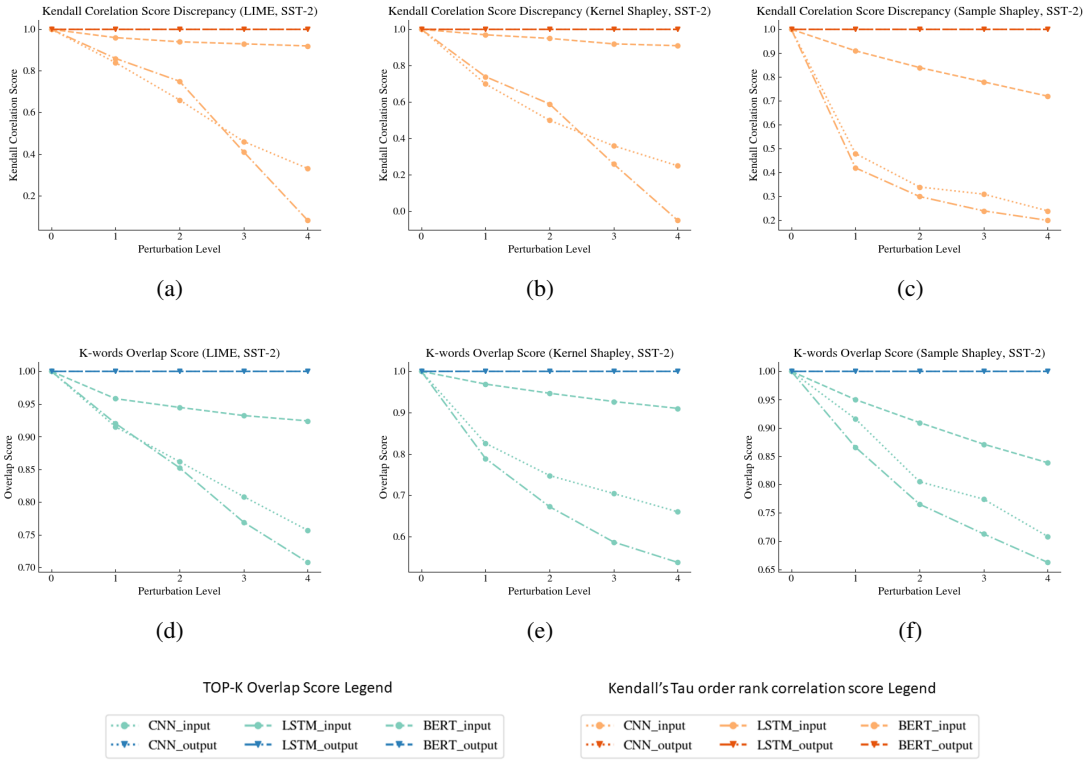

Figure 4: Comparison experiment results on the SST-2 dataset; (a) and (d) demonstrate results using LIME method; (b) and (e) demonstrate results using Kernel Sharply method; (c) and (f) demonstrate results using Sample Sharply method.

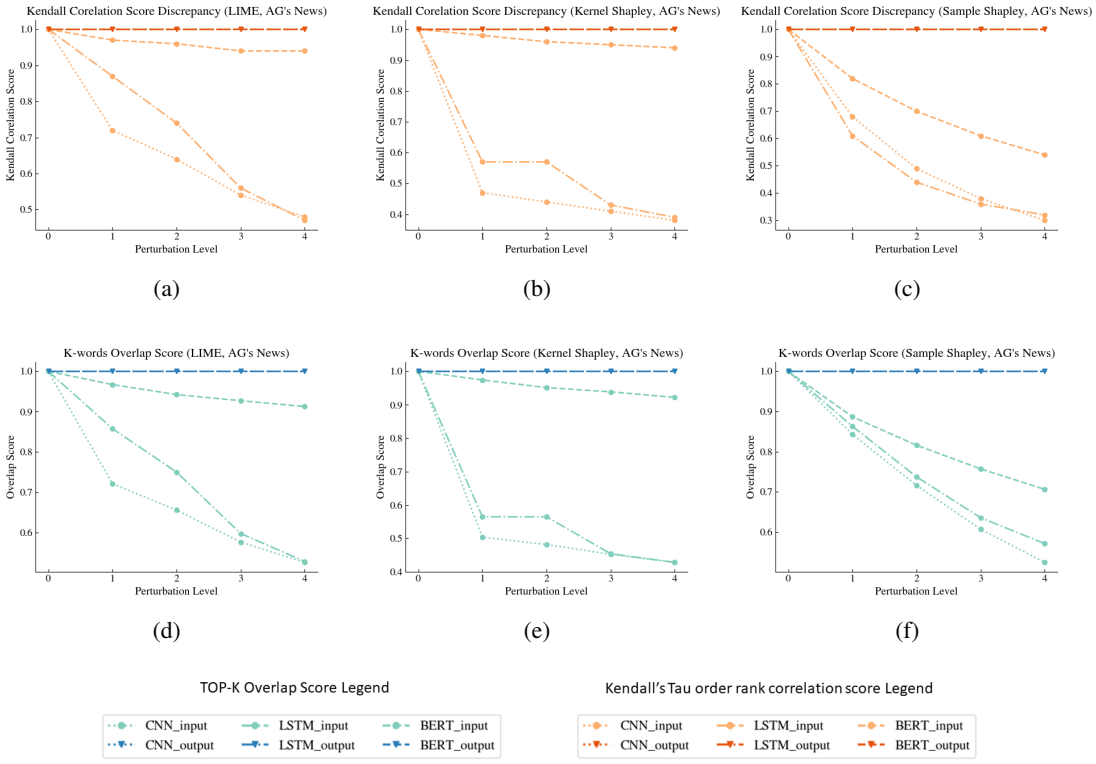

Figure 5: Comparison experiment results on the AG's News dataset; (a) and (d) demonstrate results using LIME method; (b) and (e) demonstrate results using Kernel Sharply method; (c) and (f) demonstrate results using Sample Sharply method.

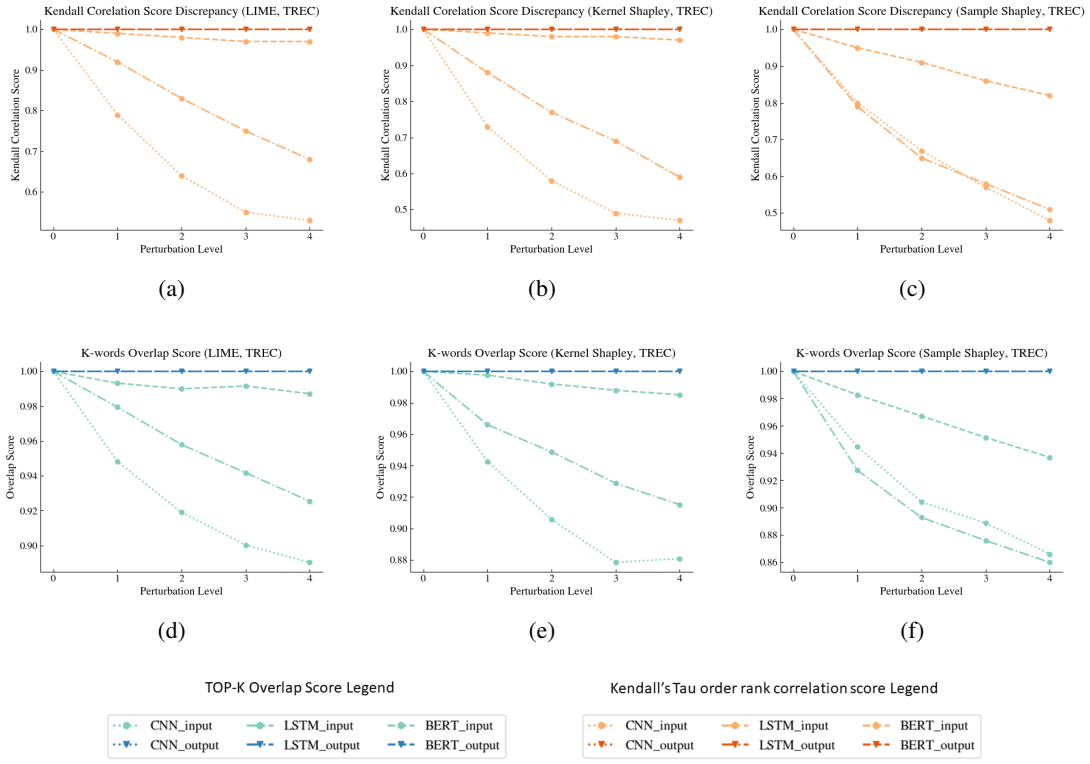

Figure 6: Comparison experiment results on the TREC dataset; (a) and (d) demonstrate results using LIME method; (b) and (e) demonstrate results using Kernel Sharply method; (c) and (f) demonstrate results using Sample Sharply method.