PFLex: Perturbation-Free Local Explanations in Language Model-Based Text Classification

Yogachandran Rahulamathavan[1,*,†], Misbah Farooq[1,†] and Varuna De Silva[1,†]

[1]Institute for Digital Technologies, Loughborough University, UK

Abstract

Large Language Models (LLMs) excel at text classification but remain difficult to interpret. Traditional methods like Local Interpretable Model-Agnostic Explanations (LIME) and Shapley Additive Explanations (SHAP) rely on input perturbations, requiring thousands of model passes, which makes them computationally expensive and unscalable for large models. To address this, we propose a structured learning framework that estimates word importance using a Siamese neural network, eliminating the need for perturbations. Our approach generates one-shot explanations, reducing computation by four orders of magnitude for BERT. Evaluated on an emotion classification and depression classification tasks, it achieves over 90% agreement with LIME. It demonstrates strong robustness, offering a scalable alternative to explain language model-based classification tasks.

1. Introduction

The adoption of LLMs in text classification has significantly improved performance across tasks such as emotion recognition, sentiment analysis, and medical diagnosis [1]. Despite their success, LLMs remain inherently opaque due to their complex architectures comprising billions of parameters. This lack of interpretability hinders trust and limits their applicability in sensitive domains like healthcare and finance [1]. Explainable AI (XAI) methods aim to enhance transparency by providing insights into model predictions [1]. The state-of-the-art XAI algorithms, such as LIME [3] and SHAP [4] approximate a model's behaviour in a localized region by analyzing the impact of perturbations on input features. For example, if the input sentence is *"I am so susceptible to feeling insecure when I see people having a good time"* then the emotion classifier identifies *sad* emotion in this sentence. Then to explain this decision, LIME assigns higher scores to the words such as *susceptible* and *insecure* in the input sentence.

As shown in Figure 1, LIME operates by systematically perturbing the input text and analyzing the resulting changes in model predictions. While effective, this approach is inherently computationally expensive, requiring thousands of perturbed inputs per explanation. The inefficiency becomes particularly problematic when applied to LLMs, where repeated inference on perturbed samples introduces significant overhead. To overcome the complexity, as shown in Figure 2, this work, named PFLex, proposes a novel deep-learning architecture which does not require perturbing the input sentence, hence significantly reducing the complexity.

1.1. Motivation

Transformers and LLMs have revolutionized NLP by using self-attention to model long-range dependencies [2]. They process sequences in parallel, capturing contextual relationships via multi-head attention. Word tokens, mapped to dense embeddings, represent semantic properties. Special tokens like [CLS] aggregate sentence information. Through transformer layers, embeddings evolve, refining the model's decisions.

Late-breaking work, Demos and Doctoral Consortium, colocated with The 3rd World Conference on eXplainable Artificial Intelligence: July 09–11, 2025, Istanbul, Turkey

y.rahulamathavan@lboro.ac.uk (Y. Rahulamathavan); M.Farooq@lboro.ac.uk (M. Farooq); V.D.De-Silva@lboro.ac.uk (V. D. Silva)

https://github.com/rahulay1 (Y. Rahulamathavan)

0000-0002-1722-8621 (Y. Rahulamathavan); 0000-0001-7535-141X (V. D. Silva)

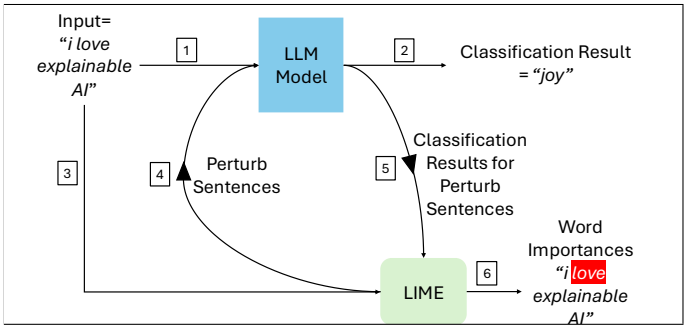

Figure 1: Block diagram illustrating LIME-based word importance estimation for text classification.

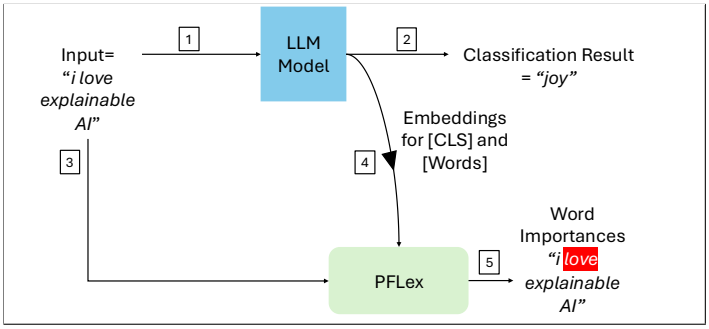

Figure 2: Block diagram illustrating the proposed PFLex approach for word importance estimation for text classification.

Intuitively, words critical to the classification task (i.e., emotion) should be more aligned with the [CLS] embedding. In the example sentence provided earlier, the distance (i.e., Euclidean) between the [CLS] embedding and the embeddings of the tokens [susceptible] and [insecure] should be smaller than the distance between [CLS] and other tokens' embeddings such as [having]. To validate this hypothesis, we randomly selected 100 sentences with different emotions. Then we obtain the LIME word importance score for all the words in each sentence. For the words with higher importance scores, we measured the Euclidean distance between the word embeddings and the [CLS] embedding at various layers of a fine-tuned BERT model. As depicted in Figure 3, in the first layer, words identified as important by LIME tend to have greater distances from the [CLS] token. However, in the final layer, the same high-importance words cluster much closer to the [CLS] token, supporting our intuition.

While this observation supports our intuition, directly applying Euclidean distance to the embeddings is insufficient to extract the word importances perfectly. The relationship between word embeddings and classification decisions is non-linear. However, we can train a neural network to capture the non-linear relationship and extract the hidden word importance scores. The aim of the neural network should be to increase the word importance score for important words and decrease the word importance score for other words.

A Siamese neural network architecture [6] is particularly well-suited for this task. By employing two identical subnetworks, the Siamese architecture processes both the [CLS] token embedding and individual word embeddings in a shared representation space. The network is trained to maximize similarity for words with high feature importance while minimizing similarity for less relevant words. To validate the idea, we evaluated our approach on fine-tuned BERT models for emotion classification [18] and depression classification [19] using the Twitter sentiment dataset[1] and Reddit Depression Dataset [2]. The experimental results show more than 90% agreement with the LIME-based word importance score while improving efficiency by four orders of magnitude for BERT model. It should be noted that the

[1]Elkomy, A. (2024). Twitter Emotion Dataset: Unveiling the Emotional Tapestry of Social Media. Available at: https://www.kaggle.com/datasets/adhamelkomy/twitter-emotion-dataset/data.

[2]Depression: Reddit Dataset (Cleaned) (2020), https://www.kaggle.com/datasets/infamouscoder/depression-reddit-cleaned.

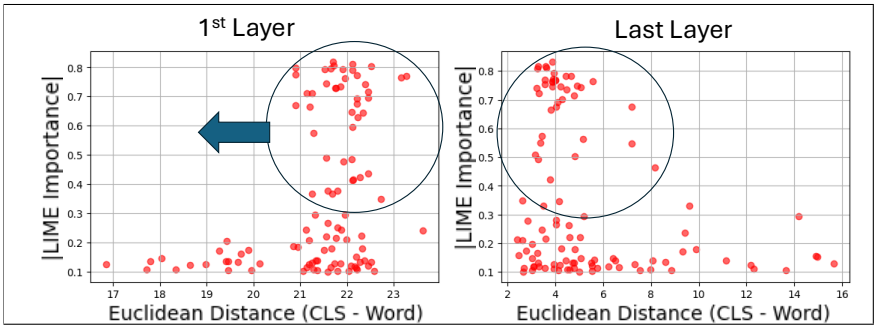

Figure 3: Comparison of Euclidean distance between [CLS] and [word] versus LIME-based word importance at different layers of BERT.

savings would increase with the larger models. Stress tests further validate its robustness, making it a scalable alternative for LLM explainability.

2. Literature Review

We review three types of XAI approaches in this section.

2.1. Feature Attribution-Based Explanations

Feature attribution methods quantify the contribution of individual input words to a model's prediction. Prominent techniques include LIME [3] and SHAP [4], which have been widely adopted in text-based decision systems [7]. These methods typically operate by perturbing inputs or applying game-theoretic principles to assess feature importance. For instance, LIME [3] generates perturbed variations of the original input by omitting or replacing words and observes how these changes affect predictions. It then fits an interpretable surrogate model to approximate the original model's decision boundary. SHAP [4], on the other hand, leverages Shapley values to attribute importance scores based on cooperative game theory, capturing both individual and interaction effects of words. Despite their effectiveness, these methods are computationally intensive. LIME and SHAP require the generation of multiple perturbed samples per input, leading to high inference costs, particularly for large-scale LLMs.

2.2. Example-Based Explanations

This approach includes adversarial examples and counterfactual explanations. As studied in [8], adversarial examples involve minimally altered inputs that cause misclassification, exposing model vulnerabilities. While valuable, these methods face challenges in computational cost and example quality. TEXTFOOLER [8], for instance, generates adversarial inputs via synonym substitution, which can be costly. Similarly, crafting meaningful counterfactual explanations requires careful input modifications to ensure interpretability. Recent advancements, such as Uni-Modal Event-Agnostic Knowledge Distillation (UEKD) [9] for multimodal fake news detection and LLM Sentinel (LLAMOS) [10] for adversarial defense, have improved the robustness of example-based explanations. Interactive XAI systems like TalkToModel [12] also enhance user understanding by facilitating human-model interactions, though ensuring explanation validity remains a challenge.

2.3. Attention-Based Explanations

Attention-based explanations leverage the inherent attention mechanisms within transformer models to provide insights into their decision-making process. These methods typically utilize attention weights to highlight influential features. For instance, [13] proposes a text classification method that combines keyword-based approaches with attention mechanisms. Similarly, AttentionViz [14] leverages attention patterns to reveal relationships within the model. However, recent research has highlighted

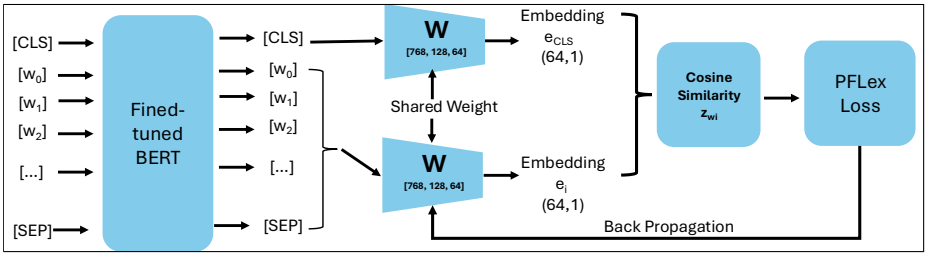

Figure 4: The training architecture diagram for the proposed PFLex approach.

the limitations of relying solely on attention weights for faithful explanations. The study in [15] has demonstrated that attention weights may not always accurately reflect the true importance of input features and can even be misleading in certain cases [16]. These limitations stem from the fact that attention weights can encode information beyond feature importance, leading to misinterpretations [17].

3. Methodology

3.1. Training Data Construction

To train a perturbation-free word importance model, we use LIME-generated scores as ground truth. Sentences are passed through a fine-tuned BERT for emotion predictions. LIME perturbs words, trains a surrogate model, and assigns importance scores. These scores are used as labels. Sentences are converted to word-CLS embedding pairs, framing the task as word similarity estimation. With approximately 1000 sentences per class and the average length of 30 words, the dataset contains around 200,000 word embeddings for emotion classification and 60,000 for depression classification.

3.2. Siamese Network Architecture and Training

Let us denote the [CLS] token embedding as $\mathbf{h}_{cls}$, word embeddings as $\mathbf{h}_w$ and word importance score as fI_w. To model the relationship between word embeddings and their importance, we employ a Siamese network [6]. As shown in Figure 4, the network consists of two identical subnetworks that transform the [CLS] token embedding and word embeddings into a shared representation space. The objective is to maximize similarity for words with high importance and minimize similarity for less relevant words. Our network consists of two fully connected layers with ReLU activation and dropout for regularization. The transformation function is given by:

$$\mathbf{e}_{cls} = \mathbf{W}(\mathbf{h}_{cls}), \quad \mathbf{e}_w = \mathbf{W}(\mathbf{h}_w). \tag{1}$$

The similarity between transformed embeddings is computed using Cosine similarity:

$$\text{sim}(\mathbf{e}_{cls}, \mathbf{e}_w) = \frac{\mathbf{e}_{cls} \cdot \mathbf{e}_w}{\|\mathbf{e}_{cls}\| \|\mathbf{e}_w\|}. \tag{2}$$

To train the network, we use GRPO. GRPO is a reinforcement learning-inspired optimization strategy designed to stabilize learning and improve convergence in policy-based learning tasks. In our context, GRPO is employed to fine-tune the Siamese network such that words with higher feature importance scores align more closely with the [CLS] token in the learned representation space. The loss function we used to train the model is defined as follows:

$$\mathcal{L} = -\sum_{i=1}^{G} fI_w^{(i)} \cdot \text{sim}(\mathbf{e}_{cls}, \mathbf{e}_w) + \lambda \sum_{i=1}^{G} \text{sim}(\mathbf{e}_{cls}, \mathbf{e}_w)^2. \tag{3}$$

where λ is a granularity factor which is selected as 1 in this context and G denotes the group of words selected for a given epoch. The first term in the loss function attempts to increase $fI_w^{(i)} \cdot \text{sim}(\mathbf{e}_{cls}, \mathbf{e}_w)$.

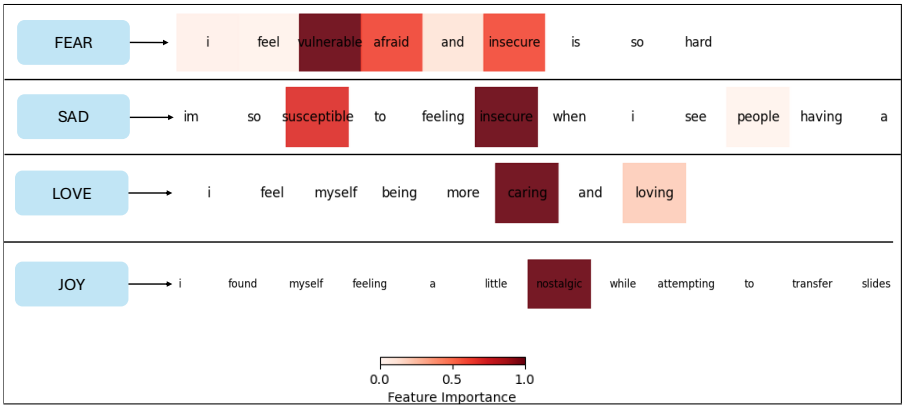

Figure 5: Feature importance obtained via the proposed PFLex method.

Therefore, if the word is important for the classification (i.e., $fI_w^{(i)}$ is positive) then the network optimises the weights such that $\text{sim}(\mathbf{e}_{cls}, \mathbf{e}_w)$ is positive. On the other hand, if the word is not important for the classification (i.e., $fI_w^{(i)}$ is negative) then the neural network weights are optimised such that the $\text{sim}(\mathbf{e}_{cls}, \mathbf{e}_w)$ is negative. Due to the requirement for both positive and negative similarity scores, Cosine similarity was employed. However, to mitigate the tendency of similarity scores to converge towards extreme values of $+1$ or -1 during loss minimization, a regularization term, $\lambda \sum_{i=1}^{G} \text{sim}(\mathbf{e}_{cls}, \mathbf{e}_w)^2$, was incorporated into the loss function (Equation 3). This term serves to discourage the attainment of maximum or minimum similarity values, thereby stabilizing the training process.

4. Experimental Setup

We validated our method using fine-tuned BERT models [18, 19] ($110M$ parameters, 12 layers, 93.4% emotion, 98.8% depression accuracy). Datasets: Twitter emotion (6 classes, $1000/200$ train/test per class, $\sim$ 30 words/Tweet) and Reddit depression (2 classes, $800/200$ train/test per class). CLS-word embedding pairs with LIME-derived importance scores (-1 to 1) were created ($\sim 200,000$ emotion, $\sim 50,000$ depression embeddings). The Siamese network (two subnetworks, $784 \rightarrow 512 \rightarrow 128 \rightarrow 64$ layers, ReLU, 20% dropout) transformed embeddings into a latent space. Cosine similarity (1) was computed and optimized using loss (3), Adam (1×10^{-4} learning rate), and 300 epochs. Feature extraction and training (6000 sentences) took $\sim$ 1 hour each on a 16GB RAM, RTX 2080 GPU system.

5. Experimental Results

To qualitatively assess our method, we generate heatmaps illustrating word importance as determined by PFLex for selected test sentences (Figure 5). The visual representation reveals a clear correspondence with PFLex's importance scores. To quantify the proposed approach, we perform a stress test comparing PFLex against LIME below.

5.1. Stress test

A stress test evaluates feature importance faithfulness by perturbing input data and observing its impact on model predictions. In text classification, this involves removing important words (e.g., by LIME or PFLex) and measuring the accuracy drop. Significant degradation upon removal indicates genuine feature importance. We performed the stress test to measure the faithfulness of the proposed PFLex approach. As shown in Figure 6, the removal of the most important features leads to a sharp decline in overall accuracy for both tasks. With all features present, the original model achieves a high accuracy of 93% and 98% for emotion and depression tasks. However, for the emotion task, when the single most important word is removed, accuracy drops drastically to 19.77% using LIME and 34.20% using

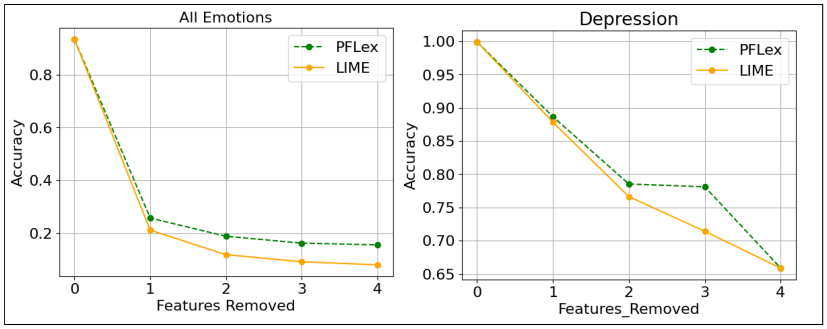

Figure 6: Overall classification accuracy for both the emotion and depression classification declines as the most important features are removed, comparing LIME-based and PFLex-based feature importance.

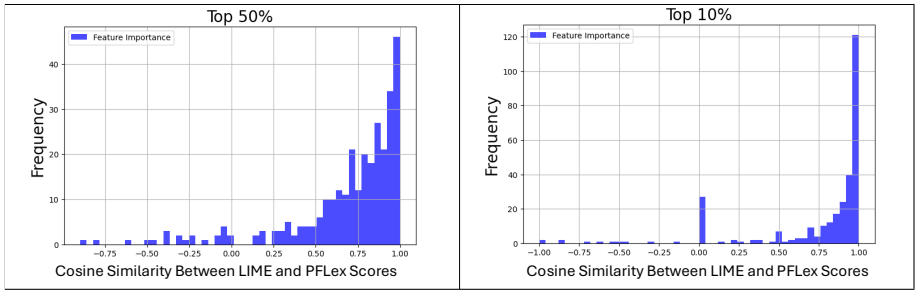

Figure 7: The histograms depict the frequency of cosine similarity measurements when retaining (1) all features, and (2) the top 10% of highly influential words between LIME and PFLex.

PFLex. This pattern continues with additional removals, reinforcing the critical role these top-ranked words play in determining model predictions. A similar pattern was observed for the depression task, validates the effectiveness of PFLex as a perturbation-free alternative to LIME.

5.2. Evaluating Alignment Between LIME and PFLex Feature Importance Scores

To assess the agreement between LIME and PFLex feature importance rankings, we computed cosine similarity scores between their word-level importance values. The analysis was conducted under varying levels of feature selection, progressively filtering out less significant words to focus on the most impactful ones. The histograms in Figure 7 illustrate how this similarity evolves across different filtering thresholds. When all features are considered, the cosine similarity between LIME and PFLex exhibits a wider spread (showing only 60% correlation), indicating moderate alignment. However, when we consider the top 10% of words with the highest absolute importance scores—the correlation reaches more than 90% between the two methods. This supports the hypothesis that PFLex effectively identifies the most crucial features in a manner that closely aligns with LIME, particularly for the most influential words in a sentence.

5.3. Complexity Comparision

Figure 8 shows the comparison between LIME and PFLex in terms of execution time and computational cost. In terms of execution time, LIME exhibits a substantial processing delay due to its perturbation-based approach. For small sentences (10 words), LIME takes approximately 5 seconds, whereas PFLex completes the explanation in just 0.52 seconds, achieving nearly a 10-time speedup. This disparity becomes even more pronounced as sentence length increases. For medium-length sentences (20 words), LIME requires 7.13 seconds, while PFLex remains highly efficient at 0.54 seconds. The most striking difference occurs for long sentences (30 words), where LIME takes an overwhelming 40.77 seconds, whereas PFLex maintains a stable processing time of just 0.55 seconds.

The computational cost analysis, shown in the second bar chart, further emphasizes the advantage of

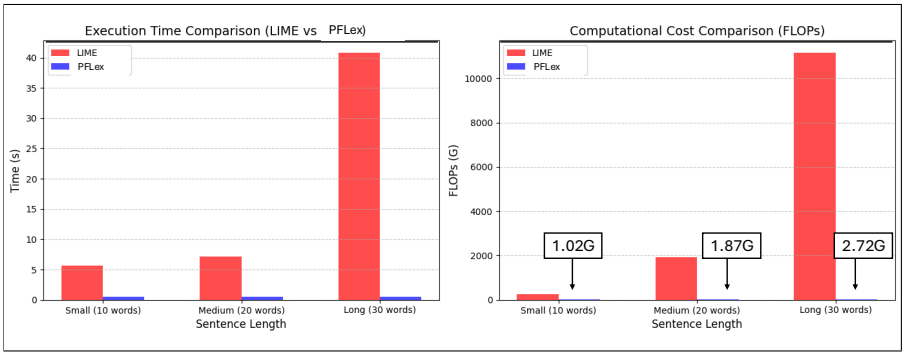

Figure 8: Comparison of Execution Time and Computational Cost for LIME vs. PFLex. (Left) Execution time comparison across different sentence lengths. (Right) Computational cost in terms of FLOPs (GigaFLOPs).

PFLex over LIME. LIME requires a substantial number of FLOPs due to the repeated inference steps needed to generate perturbed samples. For small sentences, LIME requires $261G$ FLOPs, while PFLex completes the task with just $1.02G$ FLOPs, representing a reduction of over 99% in computational complexity. This efficiency gain is even more pronounced for long sentences, where LIME demands $11,147G$ FLOPs, compared to only $2.72G$ FLOPs for PFLex. Overall, the results demonstrate that PFLex offers a significantly more scalable and computationally efficient solution compared to LIME.

6. Conclusions and Future Works

We introduced PFLex, a perturbation-free method for word-level feature importance in LLMs, using a Siamese network to directly map embeddings to importance scores. PFLex achieves LIME-comparable feature attribution with orders-of-magnitude lower computational cost. Quantitative, qualitative, and stress tests validate PFLex's effectiveness, showing high agreement with LIME and robustness. Analysis of [CLS] embeddings supports our approach's theoretical basis.

6.1. Future Works

Despite its strong performance, there remain areas for further improvement. One limitation is that PFLex relies on precomputed feature importance scores from LIME during training, which may introduce biases from perturbation-based methods. Future research will explore alternative self-supervised objectives to learn feature importance directly from the model's internal representations without requiring external supervision.

By bridging the gap between computational efficiency and interpretability, PFLex presents a promising direction for scalable, real-time explainability in LLMs. Future developments in this space could lead to even more lightweight and adaptable XAI techniques, ensuring that explainability remains accessible and practical for modern NLP applications.

Declaration on Generative AI

The author has not employed any Generative AI tools.

References

[1] Zhao et al. 2024. Explainability for large language models: A survey. ACM Trans. Intelligent Systems and Technology, 15(2), pp.1-38.

[2] Vaswani, A., 2017. Attention is all you need. Advances in Neural Information Processing Systems.

[3] Ribeiro, M.T., Singh, S. and Guestrin, C., 2016, August. "Why should i trust you?" Explaining the predictions of any classifier. In Proc. the 22nd ACM SIGKDD Int'l Conf. knowledge discovery and data mining (pp. 1135-1144).
[4] Lundberg, S., 2017. A unified approach to interpreting model predictions. arXiv preprint arXiv:1705.07874.
[5] Devlin, J., 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
[6] Bromley, J., Guyon, I., LeCun, Y., Säckinger, E. and Shah, R., 1993. Signature verification using a" siamese" time delay neural network. Advances in neural information processing systems, 6.
[7] Li, J., Zhang, Y., Karas, Z., McMillan, C., Leach, K., and Huang, Y. (2024, April). Do Machines and Humans Focus on Similar Code? Exploring Explainability of Large Language Models in Code Summarization. In Proc. the 32nd IEEE/ACM Int'l Conf. on Program Comprehension (pp. 47-51).
[8] Jin, D., Jin, Z., Zhou, J. T., and Szolovits, P. (2020, April). Is BERT really robust? a strong baseline for natural language attack on text classification and entailment. In Proc. the AAAI conference on artificial intelligence (Vol. 34, No. 05, pp. 8018-8025).
[9] Liu, G., Zhang, J., Liu, Q., Wu, J., Wu, S., and Wang, L. (2024). Uni-Modal Event-Agnostic Knowledge Distillation for Multimodal Fake News Detection. IEEE Trans. Knowledge and Data Engineering.
[10] Lin, G., and Zhao, Q. (2024). Large Language Model Sentinel: Advancing Adversarial Robustness by LLM Agent. arXiv preprint arXiv:2405.20770.
[11] Goldshmidt, R. and Horovicz, M., 2024. TokenSHAP: Interpreting Large Language Models with Monte Carlo Shapley Value Estimation. arXiv preprint arXiv:2407.10114.
[12] Slack, D., Krishna, S., Lakkaraju, H., and Singh, S. (2023). Explaining machine learning models with interactive natural language conversations using TalkToModel. Nature Machine Intelligence, 5(8), 873-883.
[13] Du, C., and Huang, L. (2018). Text classification research with attention-based recurrent neural networks. International Journal of Computers Communications and Control, 13(1), 50-61.
[14] Yeh, C., Chen, Y., Wu, A., Chen, C., Viégas, F., and Wattenberg, M. (2023). Attentionviz: A global view of transformer attention. IEEE Trans. Visualization and Computer Graphics.
[15] Shu, K., Cui, L., Wang, S., Lee, D., and Liu, H. (2019, July). defend: Explainable fake news detection. In Proc. the 25th ACM SIGKDD Int'l Conf. on knowledge discovery and data mining (pp. 395-405).
[16] Arous et al. (2021, May). Marta: Leveraging human rationales for explainable text classification. In Proc. the AAAI conference on artificial intelligence (Vol. 35, No. 7, pp. 5868-5876).
[17] Chrysostomou., G and Aletras., N. 2021. Improving the faithfulness of attention-based explanations with task-specific information for text classification. In Proc. 59th Annual Meeting of the Association for Computational Linguistics and the 11th Int'l Joint Conf. Natural Language Processing (Volume 1: Long Papers). Association for Computational Linguistics, Online, 477–488.
[18] Savani, B. (2024). Emotion Classifier. Available at: https://huggingface.co/bhadresh-savani/bert-base-uncased-emotion.
[19] Jy46604790. (2024). Fake News Detect. Available at: https://huggingface.co/jy46604790/Fake-News-Bert-Detect.