# Improving Adversarial Robustness against Sentence-level Text Attacks using Explainable AI

**Yifeng Dong**, **Karl Richter**, and **Niklas Hoelterhoff**

Department of Informatics
Technical University of Munich

## Abstract

Adversarial attacks against machine learning models in the natural language processing (NLP) sphere pose a significant security threat. A novel class of defenses aims to detect adversarial samples by using explainable AI techniques to analyze the model's reaction to a given input. Previous work has shown promising results with these methods, but have focused on token-wise explanations and attacks only. In this work, we propose methods to extend this approach to better defend against the increasingly relevant sentence level attacks, including improvements to word-level differential reaction (WDR) as well as a new neural-network specific method using layer attributions. Our experiments show that these methods can improve the detection of adversarial samples generated using sentence-level attacks.

## 1 Introduction

Deep learning techniques are advancing rapidly for many supervised learning tasks, but they often remain vulnerable to subtle changes in the input that unexplainably change the models prediction. Models with high capacity tend to learn task-irrelevant features that might not be noticable during normal operations but that attackers can take advantage of (Qi et al., 2021). By carefully perturbing an original sample, *adversarial examples* can be created that fool the victim model into predicting a wrong class with high certainty (Kurakin et al., 2016). In the image domain, original and adversarial samples are often indistinguishable for humans as perturbations are on a pixel-level (Szegedy et al., 2014; Goodfellow et al., 2014).

Adversarial attacks on text are more complex, as perturbations cannot be random but are constrained by a number of properties to maintain the validity of a sentence to the human. These include lexical, grammatical and semantic constraints. Invalid words and sentences are easily picked up by humans, thus rendering the attack invalid (Zhao et al., 2017). Current attack methods in the text domain can be clustered into three categories, based on the level of perturbation performed: Character-level attacks, word-level attacks and sentence-level attacks.

*Character-level attacks* greedily replace, swap or delete characters from the original sample (Pruthi et al., 2019; Gao et al., 2018). *Word-level attacks* such as PWWS (Ren et al., 2019) and TextFooler (Jin et al., 2019) replace words by synonyms, until the model changes its output. *Sentence-level attacks* alter the entire structure of a sentence while preserving its meaning. Common methods include style-transfer or translation between languages (Qi et al., 2021).

As there is a multitude of perturbation strategies and the victim model is usually available, many attacks achieve a high attack success rate. In contrast, the detection of adversarial examples is often challenging as changes are subtle and the attack methods are unknown. Previous work mostly focused on the detection of word-level perturbations as the attack mechanism is the most mature. In this work, we propose new methods to improve the effectiveness of defense mechanisms for samples generated using novel sentence-level attacks.

## 2 Previous Work

Current methods mitigate the risk of adversarial attacks using three approaches: Increasing the models robustness by training with adversarial samples (Yoo and Qi, 2021), pre-processing inputs to eliminate potential perturbations performed by an attacker (Alsmadi et al., 2021), or lastly, training a classifier to detect the adversarial samples (Raina and Gales, 2022).

Training with adversarial samples helps increase the robustness of a model. However, with the ever evolving adversarial attacks on texts, keeping up the model training with the latest adversarial samples becomes challenging. Pre-processing inputs is very effective against character-level attacks, since they introduce spelling errors into the text that can be easily detected (Pruthi et al., 2019), but is not suitable to detect word- and sentence-level attacks, since these do not introduce any grammatical or semantic errors into the text.

Recent methods for detecting adversarial samples have shown success by employing explainable AI (XAI) methods. These train a discriminator to distinguish between original and adversarial samples based on model attributions derived using XAI methods on the targeted neural network (see fig. 1). Current approaches are based on SHAP or saliency to derive model attributions.

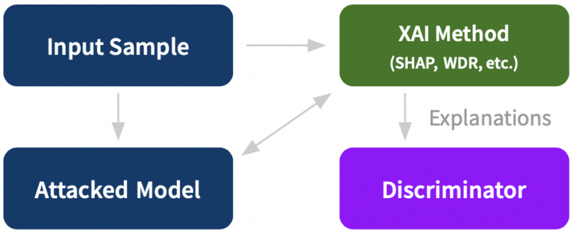


Figure 1: Overview of the explainable AI defense architecture.

**SHAP-based detectors** SHapley Additive exPlanations (Lundberg and Lee, 2017) are a commonly used method to explain contributions to model outputs of a given input. Promising results in the image domain showed the potential of using SHAP signatures for adversarial sample detection (Fidel et al., 2020), leading the way for the application in the text domain: Here, the difference between SHAP signatures of original samples and adversarial samples can also be picked up by a classification model, thus leading to a high accuracy in adversarial sample detection (Huber and Kühn, 2021).

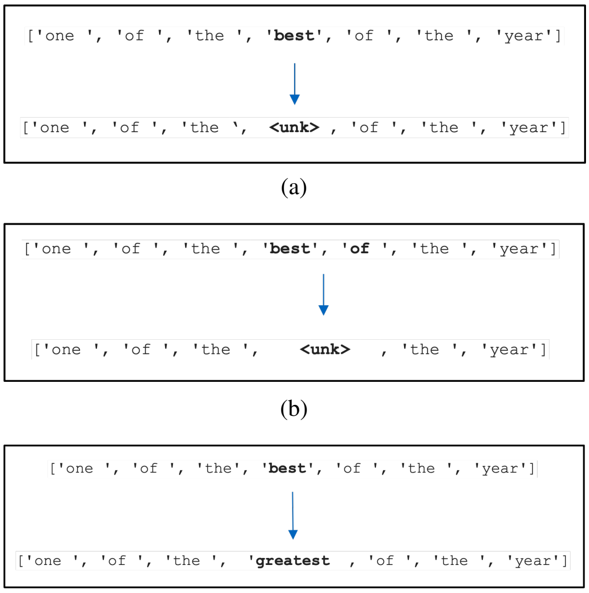
**Logit-based detectors** In previous research in the vision domain, logit analysis proved to be an effective tool for discriminating manipulated inputs in the model (Aigrain and Detyniecki, 2019; Hendrycks and Gimpel, 2016; Wang et al., 2021). Current research in the text domain demonstrates the feasibility for the detection of word-level attacks: Using a logits-based metric called Word-Level Differential Reaction (WDR), words with a suspiciously high impact on the model output can be detected (Mosca et al., 2022). In WDR, each word in the sample is successively masked (replaced with the unknown word token `unk`, fig. 2a) and the change in the model's prediction after masking is recorded. A discriminator is trained to recognize adversarial samples based on the WDR distributions.

## 3 Methodology

We propose two kinds of novel techniques to better defend against sentence-level attacks using XAI techniques. First, we propose extensions that build on WDR (Mosca et al., 2022). Second, we introduce a novel method which detects adversarial samples based on a neural network's layer attribution scores.

### 3.1 Improvements to WDR

We propose two improvements to the WDR method: masking multiple consecutive words and using a large masked language model (MLM) to fill in masked words instead of using the `unk` token.



```
['one ', 'of ', 'the ', 'best', 'of ', 'the ', 'year']
['one ', 'of ', 'the ', <unk> , 'of ', 'the ', 'year']
```

(a)

```
['one ', 'of ', 'the ', 'best', 'of ', 'the ', 'year']
['one ', 'of ', 'the ', <unk> , 'the ', 'year']
```

(b)

```
['one ', 'of ', 'the', 'best', 'of ', 'the ', 'year']
['one ', 'of ', 'the ', 'greatest , 'of ', 'the ', 'year']
```

(c)

Figure 2: Creation of masked sentences in vanilla WDR (a), with multi-word masking (b), and with language model masking (c)

**Multi-word Masking** The WDR defense analyzes the change in the model's reaction when each word in a given input sentence is successively masked. Intuitively, if a sentence has been perturbed by a word-level attack, the model should recover its original classification once the perturbed word (or one of the perturbed words) is masked out. But sentence-level attacks do not perturb input samples by only swapping out some of the words, they rather tend to alter the entire structure of the sentence. In this setting, masking out a single word may not fully mitigate the adversarial perturbation, since the perturbation may rely not only on a single word but on the entire phrase structure of the sentence.

Therefore, we propose to extend WDR by also masking multiple consecutive words at the same time. In addition to per-word scores, WDR scores are calculated for each n-gram ($n \leq N$, a hyperparameter) using the same formula. The attributions are sorted and fed to the discriminator as additional features. We hypothesize that this way the discriminator can better detect adversarial perturbations which distribute their effect across multiple words in a phrase. Only considering consecutive words means that the computational overhead is only a scalar factor at most, whilst capturing a relatively significant amount of structural information, since neighboring words are most likely to be correlated with each other.

**Language Model Masking** WDR replaces masked words with the unknown word token `unk` to create the comparison sentence. This method is simple to implement, but introduces an uncontrolled distributional bias to the comparison, since the unknown word token is not semantically neutral: rather, it signifies that the word is rare or out-of-vocabulary. This weakens the theoretical grounding of the method. The goal of WDR is to compare the input sentence to a hypothetical sentence without the masked word, which (Mosca et al., 2022) denote as $x \setminus x_m$:

$$WDR(x_m, f) = f(x \setminus x_m)_{y^*} - \max_{y \neq y^*} f(x \setminus x_m)_y$$

where $y^* = \arg\max_y f(x)_y$ is the predicted class. More specifically, the goal is that if the masked word is in fact the result of an adversarial perturbation, the comparison sentence without that word should capture the model's reaction *had the adversarial perturbation not occurred*. In vanilla WDR, $x \setminus x_m$ is formed by simply replacing $x_m$ with `unk`, but this does not actually represent a typical original sentence. Rather, the more rigorous formulation would be to compare an input $x$ with masked word $x_m$ against the **typical innocuous sentence** without the word $x^* = x_1 ... x_m^* ... x_l$ given by the natural language distribution $p(x^* | x_1 ... x_{m-1} x_{m+1} ... x_l) = p(x^* | x \setminus x_m)$.

This yields

$$\mathbb{E}_{x^* \sim p(\cdot | x \setminus x_m)} [f(x^*)_{y^*} - \max_{y \neq y^*} f(x^*)_y]$$

as a new target. It is easy to see that using `unk` to form $x^*$ does not provide an unbiased estimator for this value. Rather, we need to estimate by sampling from the conditional distribution $p(x^* | x \setminus x_m)$, e.g. by estimating with $\frac{1}{k} \sum_{i=1}^{k} f(x_i^*)_{y^*} - \max_{y \neq y^*} f(x_i^*)_y$. Sampling in this way involves solving a masked language model (MLM) problem. In our work, we use a BERT (Devlin et al., 2019) MLM for this task. To reduce computational costs, we only sample a single $x^*$ and always use the most likely word.

### 3.2 Layer-attribution adversarial detection

While black-box model-attribution methods have been shown to work well for detecting adversarial examples generated via word-level attacks, we found that they perform significantly worse for examples generated via novel sentence-level attacks. Due to the more pronounced change that sentence-level attacks perform on a sample, the model attributions seem insufficient to train a detector that discriminates well between original and adversarial samples.

We suspect that model attributions aggregate many of the fine-grained changes in activation within the language model, removing the information necessary to discriminate between original and adversarial. Thus, we hypothesized that layer-attributions could be more suitable to detect adversarial samples. Consequently, our new method extracts attributions for each token on *each layer* of a BERT model. We then either (i) concatenate all attributions, (ii) sum all attributions, or (iii) use only the layer exhibiting the strongest difference in attribution between original and adversarial examples during validation, and train a discriminator to detect adversarial samples based on the attributions.

## 4 Experiments

### 4.1 Baseline evaluation

To provide a baseline for our experiments, we first investigated the performance of naive WDR when applied to a sentence-level attack (StyleAdv (Qi et al., 2021)) and a character-level attack (Pruthi et al., 2019). In addition to training and evaluating WDR models on the new attacks, we also investigated the performance of a model trained only on word-level attacks when tested on the new attacks in a zero-shot manner (generalization performance), and vice versa. Table 1 shows the results.

| Attack | Type | Dataset | Attacked Model | F1-Score |
|---|---|---|---|---|
| *StyleAdv* | *Sentence-level* | *SST-2* | *BERT* | *0.721* |
| Pruthi | Character-level | SST-2 | BERT | 0.699 |
| PWWS | Word-level | AG-News | DistilBERT | 0.806 |
| *Pruthi* | *Character-level* | *SST-2* | *BERT* | *0.700* |
| StyleAdv | Sentence-level | SST-2 | BERT | 0.671 |
| PWWS | Word-level | AG-News | DistilBERT | 0.894 |
| *PWWS* | *Word-level* | *AG-News* | *DistilBERT* | *0.936* |
| Pruthi | Character-level | SST-2 | BERT | 0.676 |
| StyleAdv | Sentence-level | SST-2 | BERT | 0.706 |

Table 1: Baseline performance of vanilla WDR on word-level and newer attacks. The *italicized* configuration is the configuration the discriminator was trained on; the following rows in a group show the generalization performance of the discriminator on different test configurations.

**Dataset** For StyleAdv, we generated our own dataset of adversarial and original text samples consisting of 926 training and 357 validation adversarial-original pairs from the SST-2 corpus (Socher et al., 2013). For Pruthi, we used the dataset released by (Yoo et al., 2022) consisting of 610 pairs, also from the SST-2 corpus. For the word-level attack PWWS, we used the datasets from (Mosca et al., 2022) .

**Results** The results show that vanilla WDR already performs surprisingly well on the novel attacks, even in the zero-shot setting where the discriminator has only seen word-level attacks during training. As expected, performance increases further when the discriminator is trained on the novel attacks. However, vanilla WDR still performs significantly worse on the novel attacks than on traditional word-level attacks.

The fact that a word-level defense such as WDR can perform well on sentence and character-level attacks suggests that these new attacks also often concentrate their perturbative effect on individual words, instead of distributed over larger areas. It is somewhat counterintuitive that WDR performs slightly worse on the character-level than on the sentence-level attack, since one would expect that masking out the word in which a character-level perturbation took place would remove the perturbation just as effectively as with a word-level (e.g. synonym) perturbation. A possible explanation may be that character-level attacks tend to target different words as word-level attacks. For example, if character-level attacks tend to target more semantically important words in a sentence, masking out the perturbation may not allow the model to recover the original prediction as accurately. Further work would be needed to confirm our hypothesis.

### 4.2 Improvements to WDR

We tested multiple variants of our proposed improvements to WDR on StyleAdv and Pruthi using the same datasets as in 4.1. For multi-word masking, we tested variants which masked all consecutive 2, 3, or 4-grams. For MLM masking, we used BERT-Base as language model. We used a Random Forest classifier for the Pruthi discriminator and an XGBoost (Chen and Guestrin, 2016) discriminator for StyleAdv. Table 2 shows the results. Since it is non-trivial to use currently available MLMs to fill sequences of multiple words, we have not yet tested the two methods in combination. We tested the zero-shot generalization performance of multi-word masking trained on StyleAdv when evaluated on other attacks and datasets (table 3).

**Results** The results show that using multi-word masking improves performance by a small but noticeable amount for StyleAdv. The improvement increases with larger $N$ but stays constant after $N = 3$. For Pruthi, no improvement is observed. This is in line with our expectations, since masking multiple words was motivated by sentence-level attacks and masking out a single word is usually sufficient to remove a character-level perturbations.

Further, we observe that multi-word masking improves generalization performance for both character and word-level attacks quite significantly. This is a promising result and suggests that the performance increase is not simply due to overfitting. The discrepancy that multi-word masking does not yield benefits with Pruthi when directly training on the attack, but does yield a benefit in the generalization setting, warrants further examination.

Finally, we observed no benefit from using MLM masking for either attack compared to vanilla WDR using the `unk` token. This suggests that `unk` ac-

| Attack | Defense | Dataset | Attacked Model | F1-Score |
|---|---|---|---|---|
| StyleAdv | WDR | SST-2 | BERT | 0.721 |
| StyleAdv | WDR+Mask 1-2 | SST-2 | BERT | 0.728 |
| StyleAdv | WDR+Mask 1-3 | SST-2 | BERT | **0.734** |
| StyleAdv | WDR+Mask 1-4 | SST-2 | BERT | **0.734** |
| StyleAdv | WDR+BERT Mask | SST-2 | BERT | 0.714 |
| Pruthi | WDR | SST-2 | BERT | **0.700** |
| Pruthi | WDR+Mask 1-3 | SST-2 | BERT | 0.697 |
| Pruthi | WDR+BERT Mask | SST-2 | BERT | 0.676 |

Table 2: Performance of WDR with multiple token masking ($N \in \{2, 3, 4\}$) and MLM masking with a BERT language model, trained and evaluated on StyleAdv and Pruthi.

| Attack | Defense | Dataset | Attacked Model | F1-Score |
|---|---|---|---|---|
| *StyleAdv* | *WDR+Mask 1-3* | *SST-2* | *BERT* | *0.734* |
| Pruthi | WDR | SST-2 | BERT | 0.699 |
| Pruthi | WDR+Mask 1-3 | SST-2 | BERT | **0.720** |
| PWWS | WDR | AG-News | DistilBERT | 0.819 |
| PWWS | WDR+Mask 1-3 | AG-News | DistilBERT | **0.856** |

Table 3: Generalization performance of WDR with multi-word masking ($N = 3$) trained on StyleAdv (*italics*) and evaluated zero-shot on Pruthi and PWWS, compared to the baseline generalization performance (vanilla WDR trained on the same configuration).

tually works well to communicate to the model that the masked word is missing in a neutral way, that is, without adding any semantic bias. Using `unk` may in fact be *more* neutral than our method of picking the replacement word with the highest likelihood, since our method forces us to choose a single word which may have a high semantic weight, whereas `unk` always remains relatively neutral. Our method's performance may therefore be explained by insufficient sampling leading to a large variance in the estimated scores. Sampling more words may improve performance, albeit at the expense of computational cost.

### 4.3 Layer-attribution adversarial detection

To assess the effectiveness of Layer-attribution methods, we designed an experiment to quantify the accuracy of adversarial example detection for both word and sentence-level attacks.

**Dataset** For this experiment we used the same PWWS and StyleAdv datasets as in the previous experiments. We used finetuned BERT-Base models provided by textattack (Morris et al., 2020) as victim models.

**Attribution methods** To calculate the layer attributions of original and adversarial examples, we used three methods implemented by the PyTorch model interpretability library Captum (Kokhlikyan et al., 2020): Layer Activation, Layer Integrated Gradients, and Layer Gradient × Activation. Layer Activation is a simple attribution method that provides the activation of each neuron in a given layer. Layer Integrated Gradients provides the integral of the gradient with respect to the layer input along the layer activation at a given baseline (we used the zero vector as our baseline). Layer Gradient × Activation multiplies a layer's activation elementwise with the gradients of the target output with respect to the given layer.

**Adversarial discriminator** We also experimented with different discriminator model architectures, ranging from models with low (Naive Bayes) to higher capacity (Random Forest), in order to better gauge the complexity of the attribution data.

**Results** We analyzed the discriminator performance with all combinations of attribution methods and attribution aggregation strategies for the AG-News dataset and PWWS attack. The results can be found in table 4. The data shows that the Layer Activation method using the attribution of all neurons performs at an F1 score of 0.77 for StyleAdv. Of the experiments conducted in this study, this is highest detection performance we have achieved for sentence-level attacks. At the same time, we observe a poor generalization performance, suggesting that this method is very specific to a particular attack and dataset configuration (see table A3).

| Attack | Dataset | Method | Attributions | F1-Score |
|---|---|---|---|---|
| PWWS | AG-News | Layer Activation | Sum | 0.62 |
| PWWS | AG-News | Layer Gradient X Activation | Sum | 0.60 |
| PWWS | AG-News | Layer Integrated Gradients | Sum | 0.59 |
| PWWS | AG-News | Layer Activation | Max. neuron | 0.63 |
| PWWS | AG-News | Layer Gradient X Activation | Max. neuron | 0.62 |
| PWWS | AG-News | Layer Integrated Gradients | Max. neuron | 0.62 |
| PWWS | AG-News | Layer Activation | Concat | 0.65 |
| PWWS | AG-News | Layer Gradient X Activation | Concat | **0.68** |
| PWWS | AG-News | Layer Integrated Gradients | Concat | 0.66 |
| StyleAdv | SST-2 | Layer Activation | Concat | **0.77** |
| StyleAdv | SST-2 | Layer Gradient X Activation | Concat | 0.64 |
| StyleAdv | SST-2 | Layer Integrated Gradients | Concat | 0.67 |

Table 4: Results of different layer attribution methods using a Random Forest classifier

The data indicates that feeding the attributions of all neurons into the discriminator without aggregation yields a better classification performance than summing the attributions of all neurons or feeding only the attributions of a single neuron. This can be explained with information loss: If the classification model has sufficient capacity, it learns the relevant features from all attributions. Aggregating or removing attributions reduces the models performance. The results further indicate that the attributions of a single neuron contain most information to discriminate between original and adversarial as the detection performance is only slightly worse than using the attribution of all neurons. This supports the hypothesis that a single to a few neurons are activated strongly differently when feeding original and adversarial examples.

Further, we experimented with different layers of the language model to calculate the attributions. The BERT base model uses 12 layers of transformers blocks as well as an embedding layer. The data shows that the performance of the discriminator strongly depends on the layer chosen. Some layers seem better suited for adversarial sample detection than others, with the difference between the best and worst performing layers reaching up to 10%. For detailed results, see table A1.

Lastly, we experimented with different classification models to increase the detection performance. We found the random forest classifier to yield the most stable performance across different configurations, though other classifiers often performed better on particular configurations (see table A2 for detailed results). The results also indicate that the capacity of models such as Naive Bayes is too small to discriminate well.

## 5 Conclusion and Future Work

In this work, we introduced promising methods to improve adversarial defenses against novel attacks.

Our experiments on **extensions to WDR** suggest that multi-word masking is an immediately useful technique and can noticeably improve WDR performance for sentence-level attacks. On the other hand, it remains to be seen whether MLM Masking can prove useful in practice. The method enjoys a solid theoretical backing, but faces empirical challenges. Using a large MLM is relatively expensive, and if the problem lies in insufficient sampling as we hypothesize, then the computational capacity must be further increased to improve performance. In addition, the MLM itself presents a target for adversarial attacks. Nevertheless, the method provides an interesting direction for future work. Further research could focus on improving masking and comparing sentence generation strategies, possibly using paraphrasing; or on performing WDR at different layers of a neural network architecture rather than treating the model as a black box.

Our results further suggest that **layer-attribution methods** are suitable for detecting adversarial examples. While previous methods have achieved better detection metrics for word-level attacks, the proposed method outperforms the detection accuracy of previous methods on sentence-level attacks. This indicates that sentence-level attacks perturb samples on a higher level than word-level attacks, and are better detectable using layer- instead of model-attribution methods. Further, we found that often a single neuron in a specific layer is activated strongly different between original and adversarial examples. Further research is needed to understand this phenomena.

## References

Jonathan Aigrain and Marcin Detyniecki. 2019. Detecting adversarial examples and other misclassifications in neural networks by introspection.

Izzat Alsmadi, Kashif Ahmad, Mahmoud Nazzal, Firoj Alam, Ala Al-Fuqaha, Abdallah Khreishah, and Abdulelah Algosaibi. 2021. Adversarial attacks and defenses for social network text processing applications: Techniques, challenges and future research directions.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. pages 785–794. ACM.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Gil Fidel, Ron Bitton, and Asaf Shabtai. 2020. When explainability meets adversarial learning: Detecting adversarial examples using shap signatures. pages 1–8.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. pages 50–56.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples.

Dan Hendrycks and Kevin Gimpel. 2016. Early methods for detecting adversarial images.

Lukas Huber and Marc Alexander Kühn. 2021. Detecting adversarial examples in the nlp domain using shapley additive explanations.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2019. Is bert really robust? a strong baseline for natural language attack on text classification and entailment.

Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, and Orion Reblitz-Richardson. 2020. Captum: A unified and generic model interpretability library for pytorch.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2016. Adversarial machine learning at scale.

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. volume 30. Curran Associates, Inc.

John X. Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp.

Edoardo Mosca, Shreyash Agarwal, Javier Rando-Ramirez, and Georg Groh. 2022. "that is a suspicious reaction!": Interpreting logits variation to detect nlp adversarial attacks.

Danish Pruthi, Bhuwan Dhingra, and Zachary C. Lipton. 2019. Combating adversarial misspellings with robust word recognition.

Fanchao Qi, Yangyi Chen, Xurui Zhang, Mukai Li, Zhiyuan Liu, and Maosong Sun. 2021. Mind the style of text! adversarial and backdoor attacks based on text style transfer.

Vyas Raina and Mark Gales. 2022. Residue-based natural language adversarial attack detection.

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. pages 1085–1097. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew 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, Seattle, Washington, USA. Association for Computational Linguistics.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks.

Yaopeng Wang, Lehui Xie, Ximeng Liu, Jia-Li Yin, and Tingjie Zheng. 2021. Model-agnostic adversarial example detection through logit distribution learning. pages 3617–3621.

Jin Yong Yoo and Yanjun Qi. 2021. Towards improving adversarial training of nlp models.

KiYoon Yoo, Jangho Kim, Jiho Jang, and Nojun Kwak. 2022. Detection of adversarial examples in text classification: Benchmark and baseline via robust density estimation. pages 3656–3672. Association for Computational Linguistics.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. 2017. Generating natural adversarial examples.

# A Appendix

| **Layer** | **F1-Score** |
|---|---|
| Embedding Layer | 0.65 |
| Layer 1 | 0.67 |
| Layer 2 | **0.72** |
| Layer 3 | 0.68 |
| Layer 4 | **0.72** |
| Layer 5 | 0.62 |
| Layer 6 | 0.65 |
| Layer 7 | 0.68 |
| Layer 8 | 0.64 |
| Layer 9 | **0.72** |
| Layer 10 | 0.68 |
| Layer 11 | 0.64 |
| Layer 12 | 0.61 |

Table A1: Results of Random Forest classifier, trained on original and adversarial examples from the AG-News dataset generated using the PWWS attack

| **Layer** | **F1-Score** |
|---|---|
| Gaussian Naive Bayes | 0.55 |
| Bernoulli Naive Bayes | 0.59 |
| SGD Classifier | 0.60 |
| KNN Classifier | 0.62 |
| C-Support Vector Classifier | 0.62 |
| Random Forest Classifier | **0.72** |
| Extra Trees Classifier | 0.73 |
| Nu-Support Vector Classifier | **0.74** |

Table A2: Results of different classifiers using the attributions of the second BERT layer, trained on original and adversarial examples from the AG-News dataset generated using the PWWS attack

| **Attack** | **Dataset** | **F1-Score** |
|---|---|---|
| *PWWS* | *AG-News* | *0.66* |
| Pruthi | SST-2 | 0.47 |
| *StyleAdv* | *SST-2* | *0.67* |
| Pruthi | SST-2 | 0.47 |

Table A3: Generalization performance of the discriminator trained on word- and sentence-level attacks to character-level samples. Using the Integrated Gradients method of the best performing layer. The *italic* configuration indicates the samples the classifier was trained on.