"That Is a Suspicious Reaction!": Interpreting Logits Variation to Detect NLP Adversarial Attacks

Anonymous ACL submission

Abstract

Adversarial attacks are a major challenge faced by current machine learning research. These purposely crafted inputs fool even the most advanced models, precluding their deployment in safety-critical applications. Extensive research in computer vision has been carried to develop reliable defense strategies. However, the same issue remains less explored in natural language processing. Our work presents a model-agnostic detector of adversarial text examples. The approach identifies patterns in the logits of the target classifier when perturbing the input text. The proposed detector improves the current state-of-the-art performance in recognizing adversarial inputs and exhibits strong generalization capabilities across different NLP models, datasets, and word-level attacks.

1 Introduction

Despite recent advancements in *Natural Language Processing* (NLP), adversarial text attacks continue to be highly effective at fooling models into making incorrect predictions (Ren et al., 2019; Wang et al., 2019; Garg and Ramakrishnan, 2020). In particular, syntactically and grammatically consistent attacks are a major challenge for current research as they do not alter the semantical information and are not detectable via spell checkers (Wang et al., 2019). While some defense techniques addressing this issue can be found in the literature (Mozes et al., 2021; Zhou et al., 2019; Wang et al., 2019), results are still limited in performance and text attacks keep evolving. This naturally raises concerns around the safe and ethical deployment of NLP systems in real-world processes.

Previous research showed that analyzing models' logits leads to promising results in discriminating manipulated inputs (Wang et al., 2021; Aigrain and Detyniecki, 2019; Hendrycks and Gimpel, 2016). However, logits-based adversarial detectors have been only studied on computer vision applications. Our work transfers this type of methodology to the NLP domain amd its contribution can be summarized as follows:

(1) We introduce a logits-based metric called *Word-level Differential Reaction* (WDR) capturing words with a suspiciously high impact on the classifier. The metric is model-agnostic and also independent from the number of output classes.

(2) Based on WDR scores, we train an adversarial detector that is able to distinguish original from adversarial input texts preserving syntactical correctness. The approach substantially outperforms the current state of the art in NLP.

(3) We show our detector to have full transferability capabilities and to generalize across multiple datasets, attacks, and target models without needing to retrain. Our test configurations include transformers and both contextual and genetic attacks.

(4) By applying a post-hoc explainability method, we further validate our initial hypothesis—i.e. the detector identifies patterns in the WDR scores. Furthermore, only a few of such scores carry strong signals for adversarial detection.

2 Background and Related Work

2.1 Adversarial Text Attacks

Given an input sample x and a target model f, an adversarial example $x' = x + \Delta x$ is generated by adding a perturbation Δx to x such that $\arg\max f(x) = y \neq y' = \arg\max f(x')$. Although this is not required by definition, in practice the perturbation Δx is often imperceptible to humans and x' is misclassified with high confidence. In the NLP field, Δx consists in adding, removing, or replacing a set of words or characters in the original text. Unlike image attacks—vastly studied in the literature (Zhang et al., 2020) and operating in high-dimensional continuous input spaces—text perturbations need to be applied on a discrete input space. Therefore, gradient methods used for images such as FGSM (Goodfellow et al., 2014) or BIM (Kurakin et al., 2017) are not useful since they require a continuous space to perturb x. Based on the text perturbation introduced, text attacks can be distinguished into two broad categories.

Visual similarity: These NLP attacks generate adversarial samples x' that look similar to their corresponding original x. These perturbations usually create typos by introducing perturbations at the character level. DeepWordBug (Gao et al., 2018), HotFlip (Ebrahimi et al., 2018) , and VIPER (Eger et al., 2019) are well-known techniques belonging to this category.

Semantic similarity: Attacks within this category create adversarial samples by designing sentences that are semantically coherent to the original input and also preserve syntactical correctness. Typical word-level perturbations are deletion, insertion, and replacement by synonyms (Ren et al., 2019) or paraphrases (Iyyer et al., 2018). Two main types of adversarial search have been proposed. *Greedy algorithms* try each potential replacement until there is a change in the prediction (Li et al., 2020; Ren et al., 2019; Jin et al., 2020). On the other hand, *genetic algorithms* such as Alzantot et al. (2018) and Wang et al. (2019) attempt to find the best replacements inspired by natural selection principles.

2.2 Defense against Adversarial Attacks in NLP

Defenses based on spell and syntax checkers are successful against character-level text attacks (Pruthi et al., 2019; Wang et al., 2019; Alshemali and Kalita, 2019). In contrast, these solutions are not effective against word-level attacks preserving language correctness (Wang et al., 2019). We identify methods against word-level attacks belonging to two broad categories:

Robustness enhancement: The targeted model is equipped with further processing steps to not be fooled by adversarial samples without identifying explicitly which samples are adversarial. For instance, *Adversarial Training* (AT) (Goodfellow et al., 2014) consists in training the target model also on manipulated inputs. The *Synonym Encoding Method* (SEM) (Wang et al., 2019) introduces an encoder step before the target model's input layer and trains it to eliminate potential perturbations. Instead, *Dirichlet Neighborhood Ensemble* (DNE) (Zhou et al., 2020) and *Adversarial Sparse Convex Combination* (ASCC) (Dong et al., 2021) augment the training data by leveraging the convex hull spanned by a word and its synonyms.

Adversarial detection: Attacks are explicitly recognized to alert the model and its developers. Adversarial detectors were first explored on image inputs via identifying patterns in their corresponding Shapley values (Fidel et al., 2020), activation of specific neurons (Tao et al., 2018), and saliency maps (Ye et al., 2020). For text data, popular examples are *Frequency-Guided Word Substitution* (FGWS) (Mozes et al., 2021) and *learning to DIScriminate Perturbation* (DISP) (Zhou et al., 2019). The former exploits frequency properties of replaced words, while the latter uses a discriminator to find suspicious tokens and uses a contextual embedding estimator to restore the original word.

2.3 Logits-Based Adversarial Detectors

Inspecting output logits has already led to promising results in discriminating between original and adversarial images (Hendrycks and Gimpel, 2016; Pang et al., 2018; Kannan et al., 2018; Roth et al., 2019). For instance, Wang et al. (2021) trains a recurrent neural network that captures the difference in the logits distribution of manipulated samples. Aigrain and Detyniecki (2019), instead, achieves good detection performance by feeding a simple three-layer neural network directly with the logit activations.

Our work adopts a similar methodology but focuses instead on the NLP domain and thus text attacks. In this case (1) logic-based metrics to identify adversarial samples should be tailored to the new type of input and (2) detectors should be tested on currently used NLP models such as transformers (Devlin et al., 2019).

3 Methodology

The defense approach proposed in this work belongs to the category of *adversarial detection*. It defends the target model from attacks generated via word-level perturbations belonging to the *semantic similarity* category. The intuition behind the method is that the model's reaction to original- and adversarial samples is going to differ even if the inputs are similar.

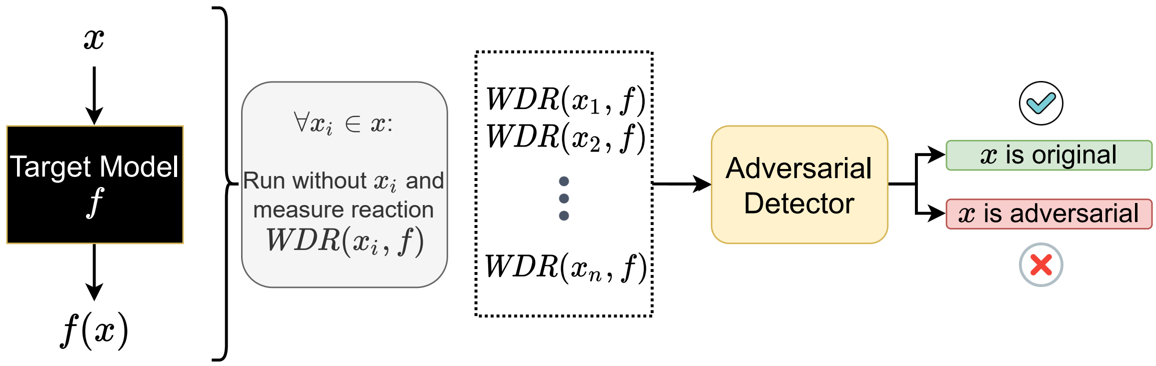

Figure 1: Overview of the proposed method.

Figure 1 shows the overall pipeline of the approach. Given a text classifier f trained on the task at hand, the pipeline's goal is to detect whether the currently fed input x is adversarial. In 3.1, we explain in greater detail how we measure the model f's reaction to a given input x. This quantity—later indicated with $WDR(x, f)$—is then passed to the adversarial detector, whose training procedure is described in 3.2. Finally, in 3.3, we provide detailed information about the setup of our experiments such as target models, datasets, and attacks.

3.1 Interpreting the Target Model and Measuring its Reaction: Word-Level Differential Reaction

Adversarial attacks based on semantic similarity replace the smallest number of words possible to change the target model's prediction (Alzantot et al., 2018). Thus, we expect the replacements transforming x into x' to play a big role for the output. If not, we would not have $f(x')$ substantially different from $f(x)$. To assess the reaction of the target model f to a given input x, we measure the impact of a word via the *Word-level Differential Reaction* (WDR) metric. Specifically, the effect of replacing a word x_i on the prediction

$$y^* = \arg\max_y p(y|x)$$

is quantified by

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

where $f(x \backslash x_i)_y$ indicates the output logit for class y for the input sample x without the word x_i. Specifically, x_i is replaced by an *unknown word token*. If x is adversarial, we could expect to find perturbed words to have a negative $WDR(x_i, f)$ as without them the input text should recover its original prediction. Table 1 shows an example pair of original and adversarial text together with their corresponding $WDR(x_i, f)$ scores. The original class is recovered after removing a perturbed word in the adversarial sentence. This switch results in a negative WDR. However, even if the most important word is removed from the original sentence (*'worst'*), the predicted class does not change and thus $WDR(x_i, f) > 0$.

Our adversarial detector takes as input $WDR(x, f)$, i.e. the sorted list of WDR scores $WDR(x_i, f)$ for all words x_i in the input sentence. As sentences vary in length, we pad the list with zeros to ensure a consistent input length for the detector.

3.2 Adversarial Detector Training

The adversarial detector is a machine-learning classifier that takes the model's reaction $WDR(x, f)$ as input and outputs whether the input x is adversarial or not. To train the model, we adopt the following multi-step procedure:

(S1) Generation of adversarial samples: Given a target classifier f, for each original sample available x, we generate one adversarial example x'. This leads to a balanced dataset containing both normal and perturbed samples. The labels used are *original* and *adversarial* respectively.

(S2) WDR computation: For each element of the mixed dataset, we compute the $WDR(x, f)$ scores as defined in Section 3.1. Once more, this step creates a balanced dataset containing $WDR(x, f)$ for both normal and adversarial samples.

Original sentence: **Negative Review (*Class 0*)**			
This is absolutely the worst trash I have ever seen. It took 15 full minutes before I realized that what I was seeing was a sick joke! [...]			
Removed x_i	**Logit *Class 0***	**Logit *Class 1***	**WDR $WDR(x_i, f)$**
$\emptyset$	3.44	-3.46	**6.89**
worst	1.68	-1.75	**3.43**
sick	3.34	-3.42	**6.76**
absolutely	3.40	-3.45	**6.86**
realized	3.41	-3.47	**6.89**

Adversarial sentence: **Positive Review (*Class 1*)**			
This is absolutely the tough trash I have ever seen. It took 15 full minutes before I realized that what I was seeing was a silly joke! [...]			
Removed x_i	**Logit *Class 0***	**Logit *Class 1***	**WDR $WDR(x_i, f)$**
$\emptyset$	-1.85	2.17	**4.02**
tough	2.14	-1.50	**-3.64**
silly	1.38	-1.37	**-2.75**
absolutely	-0.31	0.48	**0.79**
realized	-1.07	1.36	**2.43**

Table 1: $WDR(x_i, f)$ scores computed for an original sentence and its corresponding adversarial perturbation. Results show how when removing adversarial words such as *tough* or *silly*, the original class is recovered and the WDR becomes negative. $\emptyset$ corresponds to the prediction without any replacements

(S3) Detector training: The output of the second step **(S2)** is split into training and test data. Then, the training data is fed to the detector for training along with the labels defined in step **(S1)**.

Please note that no assumption on f is made. At the same time, the input of the adversarial detector—i.e. the WDR scores—does not depend on the number of output classes of the task at hand. Hence, the adversarial detector is model-agnostic w.r.t. the classification task and the classifier targeted by the attacks.

In our case, we do not pick any particular architecture for the adversarial detector. Instead, we experiment with a variety of models to test their suitability for the task. In the same spirit, we test our setting on different target classifiers, types of attacks, and datasets.

3.3 Experimental Setup

To test our pipeline, four popular classification benchmarks were used: *IMDb* (Maas et al., 2011), *Rotten Tomatoes Movie Reviews* (RTMR) (Pang and Lee, 2005), *Yelp Polarity* (YELP) (Zhang et al., 2015), and *AG News* (Zhang et al., 2015). The first three are binary sentiment analysis tasks in which reviews are classified in either *positive* or *negative* sentiment. The last one, instead, is a classification task where news articles should be identified as one of four possible topics: *World*, *Sports*, *Business*, and *Sci/Tech*.

As main target model for the various tasks we use DistilBERT (Sanh et al., 2020) fine-tuned on IMDb. We choose DistilBert—a transformer language model (Vaswani et al., 2017)—as transformer architectures are widely used in NLP applications, established as state of the art in several tasks, and generally quite resilient to adversarial attacks (Morris et al., 2020). Furthermore, we employ a *Convolutional Neural Network* (CNN) (Zhang et al., 2015), a *Long Short-Term Memory* (LSTM) (Hochreiter and Schmidhuber, 1997), and a full BERT model (Devlin et al., 2019) to test transferability to different target architectures. All models are provided by the TextAttack library (Morris et al., 2020) and are already trained[1] on the datasets used in the experiments.

We generate adversarial text attacks via four well-established word-substitution-based techniques: *Probability Weighted Word Saliency* (PWWS) (Ren et al., 2019), *Improved Genetic Algorithm* (IGA) (Jia et al., 2019), *TextFooler* (Jin et al., 2020), and *BERT-based Adversarial Examples* (BAE) (Garg and Ramakrishnan, 2020). The first is a greedy algorithm that uses word saliency and prediction probability to determine the word replacement order (Ren et al., 2019). IGA, instead, crafts attacks via mutating sentences and promoting the new ones that are more likely to cause a change in the output. TextFooler ranks words by importance and then replaces the ones with the highest ranks. Finally, BAE, leverages a BERT language model to replace tokens based on their context (Garg and Ramakrishnan, 2020). All attacks are generated using the TextAttack library (Morris et al., 2020).

[1] https://textattack.readthedocs.io/en/latest/3recipes/models.html, released under MIT License

We investigate several combinations of datasets, target models, and attacks to test our detector in a variety of configurations. Because of its robustness and well-balanced behavior, we pick the average F1-score as our main metric for detection. However, as in adversarial detection false negatives can have major consequences, we also report the recall on adversarial sentences. Later on, in 4.3, we also compare performance with other metrics such as precision and original recall and observe how they are influenced by the chosen decision threshold.

4 Experimental Results

In this section, we report the experimental results of our work. In 4.1, we study various detector architectures to choose the best performing one for the remaining experiments. In 4.2, we measure our pipeline's performance in several configurations (target model, dataset, attack) and we compare it to the current state-of-the-art adversarial detectors. While doing so, we also assess transferability via observing the variation in performance when changing the dataset, the target model, and the attack source without retraining our detector. Finally, in 4.3, we look at how different decision boundaries affect performance metrics.

4.1 Choosing a Detector Model

The proposed method does not impose any constraint on which detector architecture should be used. For this reason, no particular model has been specified in this work so far. We study six different detector architectures in one common setting. We do so in order to pick one to be utilized in the rest of the experiments. Specifically, we compare XGBoost (Chen and Guestrin, 2016), AdaBoost (Schapire, 1999), LightGBM (Ke et al., 2017), SVM (Hearst et al., 1998), Random Forest (Breiman, 2001), and a Perceptron NN (Singh and Banerjee, 2019). All models are compared on adversarial attacks generated with PWWS from IMDb samples and targeting a DistilBERT model fine-tuned on IMDb. A balanced set of $3,000$ instances—$1,500$ normal and $1,500$ adversarial—was used for training the detectors while the test set contains a total of 1360 samples following the same proportions.

As shown in Table 2, all architectures achieve competitive performance and none of them clearly appears superior to the others. We pick XGBoost (Chen and Guestrin, 2016) as it exhibits the best F1-score. The main hyperparameters utilized are 29 gradient boosted trees with a maximum depth of 3 and 0.34 as learning rate. We utilize this detector architecture for all experiments in the following sections.

Model	F1-Score	Adv. Recall
XGBoost	**92.4**	95.2
AdaBoost	91.8	**96.0**
LightGBM	92.0	93.7
SVM	92.0	94.8
Random Forest	91.5	93.7
Perceptron NN	90.4	88.1

Table 2: Performance comparison of different detector architectures on IMDb adversarial attacks generated with PWWS and targeting a DistilBERT transformer.

4.2 Detection Performance

Tables 3a and 3b report the detection performance of our method in a variety of configurations. In each table, the first row represents the setting—i.e. combination of target model, dataset, and attack type—in which the detector was trained. The remaining rows, instead, are w.r.t. settings in which we tested the already trained detector without performing any kind of fine-tuning or retraining.

We utilize a balanced training set of size $3,000$ and $2,400$ samples respectively for the detectors trained on IMDb adversarial attacks (Table 3a) and on AG News attacks (Table 3b). All results are obtained using balanced test sets containing 500 samples. The only exceptions are the configurations (DistilBERT, RTMR, IGA) and (DistilBERT, AG News, IGA) which used test sets of size 480 and 446 respectively due to data availability.

To the best of our knowledge, the FGWS method from Mozes et al. (2021) is the best detector available and was already proven to be better than DISP (Zhou et al., 2019) by its authors. Hence, we utlize FGWS as baseline for comparison in all configurations. Analogously to our method, FGWS is trained on the configuration in the first row of each table and then applied to all others. More in detail, we fine-tune its *frequency substitution threshold* parameter δ (Mozes et al., 2021) until achieving a best fit value of $\delta = 0.9$ in both training settings.

From what can be seen in both tables, the proposed method consistently shows very competi-

Configuration			***WDR*** (Ours)		***FGWS*** (Mozes et al., 2021)	
Model	**Dataset**	**Attack**	**F1-Score**	**Adv. Recall**	**F1-Score**	**Adv. Recall**
DistilBERT	IMDb	PWWS	**92.1 ± 0.5**	94.2 ± 1.1	89.5	82.7
LSTM	IMDb	PWWS	**84.1 ± 3.4**	86.8 ± 8.5	80.0	69.6
CNN	IMDb	PWWS	84.3 ± 3.1	90.0 ± 6.2	**86.3**	79.6
BERT	IMDb	PWWS	**92.4 ± 0.7**	92.5 ± 1.8	89.8	82.7
DistilBERT	AG News	PWWS	**93.1 ± 0.6**	96.1 ± 2.2	89.5	84.6
DistilBERT	RTMR	PWWS	74.1 ± 3.1	85.1 ± 8.6	**78.9**	67.8
DistilBERT	IMDb	TextFooler	**94.2 ± 0.8**	97.3 ± 0.9	86.0	77.6
DistilBERT	IMDb	IGA	**88.5 ± 0.9**	95.5 ± 1.3	83.8	74.8
DistilBERT	IMDb	BAE	**88.0 ± 0.9**	96.3 ± 1.0	65.6	50.2
DistilBERT	RTMR	IGA	**70.4 ± 5.5**	90.2 ± 6.9	68.1	55.2
DistilBERT	RTMR	BAE	**68.5 ± 4.3**	82.2 ± 9.0	29.4	18.5
DistilBERT	AG News	BAE	**81.0 ± 4.3**	95.4 ± 3.8	55.8	44.0
BERT	YELP	PWWS	89.4 ± 0.6	85.3 ± 1.7	**91.2**	85.6
BERT	YELP	TextFooler	**95.9 ± 0.3**	97.5 ± 0.6	90.5	84.2

(a) Performance results for detector trained on (DistilBERT, IMDb, PWWS).

Configuration			***WDR*** (Ours)		***FGWS*** (Mozes et al., 2021)	
Model	**Dataset**	**Attack**	**F1-Score**	**Adv. Recall**	**F1-Score**	**Adv. Recall**
DistilBERT	AG News	PWWS	**93.6 ± 1.5**	94.8 ± 2.4	89.5	84.6
LSTM	AG News	PWWS	**94.0 ± 1.0**	94.2 ± 2.2	88.9	84.9
CNN	AG News	PWWS	**91.1 ± 1.4**	91.2 ± 2.6	90.6	87.6
BERT	AG News	PWWS	**92.5 ± 0.9**	93.0 ± 1.8	88.7	83.2
DistilBERT	IMDB	PWWS	**91.4 ± 0.6**	93.0 ± 1.9	89.5	82.7
DistilBERT	RTMR	PWWS	75.8 ± 0.9	78.5 ± 4.8	**78.9**	67.8
DistilBERT	AG News	TextFooler	**95.7 ± 0.7**	97.3 ± 1.2	87.0	79.4
DistilBERT	AG News	BAE	**86.4 ± 1.1**	94.5 ± 1.8	55.8	44.0
DistilBERT	AG News	IGA	**86.7 ± 1.5**	93.6 ± 2.1	68.6	58.3
DistilBERT	RTMR	IGA	**73.7 ± 1.5**	85.4 ± 5.2	68.1	55.2
DistilBERT	RTMR	BAE	**71.0 ± 1.1**	75.2 ± 6.0	29.4	18.5
DistilBERT	IMDB	BAE	**88.1 ± 0.9**	97.0 ± 1.0	65.6	55.2
BERT	YELP	PWWS	86.2 ± 1.4	77.2 ± 3.1	**91.2**	85.6
BERT	YELP	TextFooler	**95.4 ± 0.3**	94.7 ± 0.9	90.5	84.2

(b) Performance results for detector trained on (DistilBERT, AG News, PWWS).

Table 3: Adversarial detection performance of our defense against the state of the art *FGWS* under several setups. Results were obtained with a detector trained on two different configurations as indicated in the first row of each table. For all other rows, i.e. test configurations, differences w.r.t the training setup have been highlighted. To increase the results' statistical significance, we average the performance across 30 different data-splits of the training configuration. Additionally, we report the corresponding 95% confidence intervals. Given the deterministic nature of *FGWS*, different data-splits lead to the same performance and hence confidence intervarls are not reported as they are trivial (±0).

tive results in terms of F1-score and outperforms the baseline in 22 configurations out of 28 (worse in 5) and is on average better by 8.96 percentage points. At the same time, our methods exhibits a very high adversarial recall, showing a strong capability at identifying attacks and thus producing a small amount of false negatives.

Generalization to different target models: Starting from the training configurations, we vary the *target model* while maintaining the other components fixed (rows 2-4 of each table). Here, the detector achieves state-of-the-art results in all test settings, occasionally dropping below the 90% F1-score on a few simpler models like LSTM and CNN while not exhibiting any decay on more complex models like BERT.

Generalization to different datasets: Analogous to the previous point, we systematically substitute the *dataset* component for evaluation (rows 5-6 of each table). We notice a substantial decay in F1-score when testing with RTMR (74.1 - 75.8%) since samples are short and, therefore, may contain few words which are very relevant for the prediction, just like adversarial replacements. Nevertheless, removing adversarial words still result in a change of prediction to the original class thereby preserving high adversarial recall."

Generalization to different attacks: Results highlight a good reaction to all other text attacks (rows 7-9 of each table) and even experiences a considerable boost in performance against TextFooler. In contrast, the baseline *FGWS* significantly suffers against more complex attacks such as BAE, which generates context-aware perturbation.

Besides testing generalization properties via systematically varying one configuration component at the time, we also test on a few settings presenting changes in multiple ones (rows 10-14 of each table). Also in these settings, the proposed method maintains a very competitive performance, with noticeable drops only on the RTMR dataset.

4.3 Tuning the Decision Boundary

Depending on the application in which the detector is used to monitor the model and detect malicious input manipulations, different performance metrics can be taken into account to determine whether it is safe to deploy the model. For instance, in a very safety-critical application where successful attacks lead to harmful consequences, *adversarial recall* becomes considerably more relevant as a metric than the F1-score.

We examine how relevant metrics change in response to different choices for the discrimination threshold. Please note that a lower value corresponds to more caution, i.e. we are more likely to output that a certain input is adversarial.

Figure 2 and Table 4 show performance results w.r.t. different threshold choices. We notice that decreasing its value from 0.5 to 0.15 can increase the adversarial recall to over 98% at a small cost in terms of precision and F1-score ($<$ 2 percentage points). Applications where missing attacks—i.e. false negatives—have disastrous consequences could take advantage of this property and consider lowering the decision boundary. This is particularly true if attacks are expected with a low frequency and an increase in false positive incurs only minor costs.

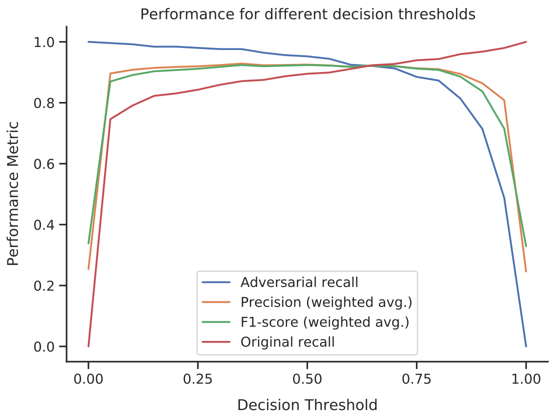

Figure 2: Performance metrics w.r.t. different decision thresholds for our XGBoost classifier on the configuration (IMDb, DistilBERT, PWWS). Input sentences are classified as adversarial when their probability is higher than the decision threshold.

DT	Precision	F1	Adv. Recall	Orig. Recall
0.5	92.5	92.4	95.2	89.5
0.4	92.3	92.0	96.4	87.5
0.3	92.4	91.8	97.6	85.9
0.15	91.5	90.3	**98.4**	82.3

Table 4: Performance comparison using different decision thresholds (DT) for our XGBoost classifier on the configuration (IMDb, DistilBERT, PWWS). The used default value is 0.5.

5 Discussion and Qualitative Results

5.1 Understanding the Adversarial Detector

The proposed pipeline consists of a machine learning classifier—e.g. XGBoost—fed with the model's WDR scores. The intuition behind the approach is that words replaced by adversarial attacks play a big role in altering the target model's decision. Despite the competitive detection performance, the detector is itself a learning algorithm and we cannot determine with certainty what patterns it can identify.

To validate our original hypothesis, we apply a popular explainability technique—SHAP (Lundberg and Lee, 2017)—to our detector. This allows us to summarize the effect of each feature at the dataset level. We use the official implementation[2] to estimate the importance of each WDR and use a *beeswarm plot* to visualize the results.

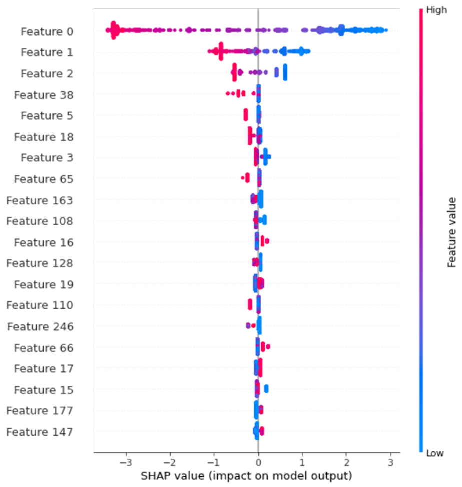

Figure 3: WDR scores with the highest impact (SHAP value) on the detector's prediction. Feature $n-1$ corresponds to the n-th WDR. For instance, feature 0 is the first and largest WDR score.

Figure 3 shows that values in the first positions—i.e. 0, 1, and 2—of the input sequence are those influencing the adversarial detector the most. Since in our pipeline WDR scores are sorted based on their magnitude, this means that the largest WDR of each prediction are the most relevant for the detector. This is consistent with our hypothesis that replaced words substantially change output logits and thus measuring their variation is effective for detecting input manipulations. As expected, negative values for the WDR correspond to a higher likelihood of the input being adversarial.

We also notice that features after the first three do not appear in the naturally expected order. We believe this is the case as for most sentences it is sufficient to replace two-three words to generate an adversarial sample. Hence, in most cases, only a few WDR scores carry important signals for detection.

[2]https://github.com/slundberg/shap, released under MIT License

5.2 Challenges and Limitations

While WDR scores contain rich patterns to identify manipulated samples, they are also relatively expensive to compute. Indeed, we need to run the model once for each feature—i.e. each word—in the input text. While this did not represent a limitation for our use-cases and experiments, we acknowledge that it could result in drawbacks when input texts are particularly long.

Our method is specifically designed against word-level attacks and it does not cover character-level ones. However, the intuition seems to some extent applicable also to sentences with typos and similar artifacts as the words containing them will play a big role for the prediction. This, like in the word-level case, needs to happen in order for the perturbations to result in a successful adversarial text attack and change the target model's prediction

6 Conclusion

Adversarial text attacks are a major obstacle to the safe deployment of NLP models in high-stakes applications. However, although manipulated and original samples appear indistinguishable, interpreting the model's reaction can uncover helpful signals for adversarial detection.

Our work utilizes logits of original and adversarial samples to train a simple machine learning detector. WDR scores are an intuitive measure of word relevance and are effective for detecting text components having a suspiciously high impact on the output. The detector does not make any assumption on the classifier targeted by the attacks and can be thus considered model-agnostic.

The proposed approach achieves very promising results, considerably outperforming the previous state-of-the-art in word-level adversarial detection. Experimental results also show the detector to possess remarkable generalization capabilities across different target models, datasets, and text attacks without needing to retrain. These include transformer architectures such as BERT and well-established attacks such as PWWS, genetic algorithms, and context-aware perturbations.

We believe our work sets a strong baseline on which future research can build to develop better defense strategies and thus promoting the safe deployment of NLP models in practice. We release our code to the public to facilitate further research and development [3].

[3]GitHub URL hidden to keep anonymity, released upon

References

Jonathan Aigrain and Marcin Detyniecki. 2019. Detecting adversarial examples and other misclassifications in neural networks by introspection. *arXiv preprint arXiv:1905.09186*.

Basemah Alshemali and Jugal Kalita. 2019. Towards mitigating adversarial texts. *International Journal of Computer Applications*, 178(50):1–7.

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896. Association for Computational Linguistics.

Leo Breiman. 2001. Random forests. *Machine Learning*, 45(1):5–32.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794. Association for Computing Machinery.

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. Association for Computational Linguistics.

Xinshuai Dong, Anh Tuan Luu, Rongrong Ji, and Hong Liu. 2021. Towards robustness against natural language word substitutions. In *9th International Conference on Learning Representations (ICLR)*.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36. Association for Computational Linguistics.

Steffen Eger, Gözde Gül Şahin, Andreas Rücklé, Ji-Ung Lee, Claudia Schulz, Mohsen Mesgar, Krishnkant Swarnkar, Edwin Simpson, and Iryna Gurevych. 2019. Text processing like humans do: Visually attacking and shielding NLP systems. 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 1634–1647. Association for Computational Linguistics.

G. Fidel, R. Bitton, and A. Shabtai. 2020. When explainability meets adversarial learning: Detecting adversarial examples using shap signatures. In *2020 International Joint Conference on Neural Networks (IJCNN)*, 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. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56.

Siddhant Garg and Goutham Ramakrishnan. 2020. Bae: Bert-based adversarial examples for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6174–6181.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. In International Conference on Learning Representations (ICLR).

M.A. Hearst, S.T. Dumais, E. Osuna, J. Platt, and B. Scholkopf. 1998. Support vector machines. *IEEE Intelligent Systems and their Applications*, 13(4):18–28.

Dan Hendrycks and Kevin Gimpel. 2016. Early methods for detecting adversarial images. *arXiv preprint arXiv:1608.00530*.

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

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885. Association for Computational Linguistics.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4129–4142. Association for Computational Linguistics.

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. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8018–8025.

Harini Kannan, Alexey Kurakin, and Ian Goodfellow. 2018. Adversarial logit pairing. *arXiv preprint arXiv:1803.06373*.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu.

acceptance

2017. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2017. Adversarial examples in the physical world. *ICLR Workshop*.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. BERT-ATTACK: Adversarial attack against BERT using BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6193–6202. Association for Computational Linguistics.

Scott M Lundberg and Su-In Lee. 2017. 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 L. 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. Association for Computational Linguistics.

John 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. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126.

Maximilian Mozes, Pontus Stenetorp, Bennett Kleinberg, and Lewis Griffin. 2021. Frequency-guided word substitutions for detecting textual adversarial examples. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 171–186. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124. Association for Computational Linguistics.

Tianyu Pang, Chao Du, Yinpeng Dong, and Jun Zhu. 2018. Towards robust detection of adversarial examples. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 4584–4594, Red Hook, NY, USA. Curran Associates Inc.

Danish Pruthi, Bhuwan Dhingra, and Zachary C. Lipton. 2019. Combating adversarial misspellings with robust word recognition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5582–5591. Association for Computational Linguistics.

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. Association for Computational Linguistics.

Kevin Roth, Yannic Kilcher, and Thomas Hofmann. 2019. The odds are odd: A statistical test for detecting adversarial examples. In *International Conference on Machine Learning*, pages 5498–5507. PMLR.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Robert E. Schapire. 1999. A brief introduction to boosting. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'99, page 1401–1406. Morgan Kaufmann Publishers Inc.

Jaswinder Singh and Rajdeep Banerjee. 2019. A study on single and multi-layer perceptron neural network. In *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)*, pages 35–40.

Guanhong Tao, Shiqing Ma, Yingqi Liu, and Xiangyu Zhang. 2018. Attacks meet interpretability: Attribute-steered detection of adversarial samples. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 7728–7739. Curran Associates Inc.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Xiaosen Wang, Hao Jin, and Kun He. 2019. Natural language adversarial attacks and defenses in word level. *arXiv preprint arXiv:1909.06723*.

Yaopeng Wang, Lehui Xie, Ximeng Liu, Jia-Li Yin, and Tingjie Zheng. 2021. Model-agnostic adversarial example detection through logit distribution learning. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 3617–3621.

Dengpan Ye, Chuanxi Chen, Changrui Liu, Hao Wang, and Shunzhi Jiang. 2020. Detection defense against adversarial attacks with saliency map. *arXiv preprint arXiv:2009.02738*.

Wei Emma Zhang, Quan Z. Sheng, Ahoud Alhazmi, and Chenliang Li. 2020. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Trans. Intell. Syst. Technol.*, 11(3).

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 649–657. MIT Press.

Yi Zhou, Xiaoqing Zheng, Cho-Jui Hsieh, Kai-wei Chang, and Xuanjing Huang. 2020. Defense against adversarial attacks in nlp via dirichlet neighborhood ensemble. *arXiv preprint arXiv:2006.11627*.

Yichao Zhou, Jyun-Yu Jiang, Kai-Wei Chang, and Wei Wang. 2019. Learning to discriminate perturbations for blocking adversarial attacks in text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4904–4913. Association for Computational Linguistics.