Semantic Sabatoge: Adversarial Attacks on Natural Language Classification

Abstract

Adversarial attacks involve deliberately modifying input to a machine learning model in a way that appears unchanged to the human eye, yet causes the model to alter its output. While these attacks are common in image classification, their application in natural language processing (NLP) is less understood. This paper aims to bridge this gap by developing a scoring function to identify important words in a given input text and strategically constructing adversarial attacks by manipulating these words. Targeting key words that influence model predictions allows the text to retain most of its original words while still preserving the semantic integrity of the text. We leverage different embedding techniques (Baseline, GloVe) and deep learning models (LSTM, CNN) to execute these attacks. This strategy offers new insights into the vulnerabilities of NLP models.

Keywords: Adversarial attacks, Adversarial Robustness, NLP, LSTM, CNN, Embedding, GloVe

1. Introduction

Deep learning has become a cornerstone in natural language processing (NLP), revolutionizing text classification tasks with unparalleled accuracy and efficiency. These tasks includes binary classifications, such as fake news detection and sentiment analysis. However, the rapid rise of deep learning models brings with it the risk of adversarial attacks.

Adversarial attacks involve the deliberate manipulation of input data to deceive models into making incorrect predictions. In text classification, these attacks may involve altering specific words or phrases in a way that changes the model's output while preserving the original meaning of the text. Such attacks compromise the reliability of NLP systems and present significant challenges for their real-world application.

In our research, we have chosen to attack Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) models (as they're appropriate to the Natural Language setting), which have been trained on different embedding spaces. Our attack strategy utilizes the nearest neighbors and Euclidean distance to perturb words and achieve desired outcomes. By uncovering the vulnerabilities of NLP models to adversarial attacks, this research aims to develop strategies for assessing and mitigating these risks. Through evaluating different models and embedding techniques, we strive to identify methods that offer greater resilience to adversarial attacks, ultimately contributing to the creation of more robust and reliable text classification systems.

In particular, our study finds that White-box attacks, which exploit internal model information to craft targeted perturbations, play a major role in undermining the accuracy of NLP models. These attacks are particularly effective, resulting in substantial decreases in model accuracy. This highlights the importance of focusing on strategies to combat White Box attacks and protect NLP models against such threats.

2. Related Work

Most studies on adversarial methods have assumed a white-box setting [3, 4]. In white-box scenarios, attackers have full access to the model giving them the ability to compute gradients. An example of this sort of attack is the Fast-Gradient Signed method [6]. Conversely, in the Black-box scenario, the attackers can only see what comes in and out of the model (hence the name Black-box), making it difficult for them to generate adversarial inputs.

In their paper "Black-box Generation of Adversarial Text Sequences to Evade Deep Learning Classifiers" [1], Gao et al. introduce the DeepWordBug algorithm, which is a method for attacking text classifiers in the black-box scenario. By employing novel scoring strategies, DeepWordBug effectively identifies critical tokens for modification and applies simple character-level transformations like swapping, substitution, deletion, and insertion to minimize the edit distance while altering the model's classification. The algorithm demonstrates remarkable performance in reducing the accuracy of state-of-the-art deep-learning models such as Word-LSTM and Char-CNN, while also showcasing the transferability of adversarial samples across different models. Moreover the method works without needing any access to the model, thereby making it applicable to many real-life scenarios.

However, a notable limitation of the approach presented in the paper is the lack of a solid mathematical foundation for the attack strategy. The simplicity of the transformation methods employed, while effective, opens the door to potential solutions such as the integration of spell checkers and autocorrects.

In the second related work, ”Adversarial Examples for Natural Language Classification Problems” [2], the authors aim to maximize an adversarial function while keeping the changes within a specified distance from the original input. The method uses an iterative approach to replace words in a sentence with their synonyms, aiming to preserve the original meaning while maximizing the adversarial function. The issue with this approach is that solving such a constrained optimization problem requires quite sophisticated methods.

Contributions: Our contribution to the field is two-fold. Firstly, we remedy [1]'s weakness of being susceptible to auto corrects, by changing the method of replacement for targeted tokens. Namely, a word won't be subject to character transformations (which can overturned with a spell-check), and instead be replaced by its synonym. Secondly, we offer a simplification of [2]'s approach by borrowing [1]'s scoring system as a means to select a fixed proportion of words to perturb from a given text. In fact, we extend [1]'s scoring system by developing a mechanism for scoring words that rest on white-box assumptions, which generate adversarial attacks that far outperform those generated using [1]'s Black-box method.

3. Adversarial Examples For Natural Language Classification

Adversarial attacks, a formidable challenge across machine learning domains, is particularly difficult in text-based scenarios. Unlike image-based attacks, where alterations may be visually subtle, text perturbations must navigate the complexities of language semantics and syntax, demanding a nuanced understanding of linguistic structures and contextual meanings. In contrast, image perturbations often involve manipulating individual pixels or adding noise, a process that doesn't require the same level of semantic understanding as text perturbations. The second meaningful difference between image and text perturbation is that the inputs in the latter are discrete. In image perturbation the inputs are the values of the pixels. Since pixel values are continuous variables, it becomes very easy for us to define a metric for the difference between the original image and it's perturbed version (for example, the L2 norm).

In the realm of adversarial attacks, adversaries may possess varying degrees of knowledge regarding the model they aim to deceive, ranging from minimal to comprehensive understanding. In the Black-box scenario, adversaries are restricted to querying the target classifier without access to the underlying model architecture. Conversely, in the White-box scenario, adversaries are granted access to the model, its parameters, and the feature set of inputs.

Therefore, we present a method for generating adversarial modifications on input tokens in the Black and White box scenarios. Considering the vast search space encompassing all potential alterations across input words and characters, we propose a two-step approach for crafting adversarial samples under both paradigms:

1. Identify words important for classification
2. Alter the important words

Algorithm 1 Algorithm for the attack

```
function EUCLID($x, y$)
    Input: Two vectors $x$ and $y$
    Output: Euclidean Distance between vectors $x$ and $y$
end function
function SCORE(Text)
    Input: Vectors of tokens $(x_i)$ in Text
    Output: List of Combined scores of each $(x_i)$ in Text
end function
function ATTACK
    $\lambda$ = k-th percentile of SCORE(Text)
    for $x_i$ in Text do
        if $x_i \geq \lambda$ then
            $x_i' = \arg\min \text{EUCLID}(x_i', x_i)$
        end if
    end for
    return Text
end function
```

3.1. Scoring

To pinpoint important tokens, we utilize scoring methods inspired by those outlined in [1], customized to align with our objectives of designing both black-box and white-box attacks. These scoring algorithms evaluate the significance of each word, in terms of the word's impact on the model's classification.

3.1.1. Black-Box Scoring:

Temporal Head Score (THS): THS evaluates the significance of the i-th token in an input sequence by measuring the difference between the model's prediction score of the sequence of tokens up to and including the ith token and the model's prediction score of the sequence of tokens up to and excluding the ith token.

$$THS(x_i) = F(x_1, x_2, \ldots, x_{i-1}, x_i) - F(x_1, x_2, \ldots, x_{i-1})$$

(where F() represents a forward pass of the model)

This allows for efficient calculation of the importance of each token in the sequence through a single forward pass in RNN models. The logic behind this method for scoring the importance of a token is that since RNNs process input tokens sequentially, any change in the evaluation of a sequence of tokens after the addition of a token will be due to that token.

Temporal Tail Score (TTS): THS evaluates tokens solely based on their preceding context, overlooking the influence of subsequent tokens. To address this limitation, we introduce the Temporal Tail Score (TTS). TTS complements THS by considering the influence of tokens following a specific token. It calculates the difference in prediction scores between two trailing parts of a sentence: one containing the token in question and one without it.

$$TTS(x_i) = F(x_i, x_{i+1}, \ldots, x_n) - F(x_{i+1}, x_{i+2}, \ldots, x_n)$$

This comparison provides insight into whether the token influences the final prediction when coupled with subsequent tokens.

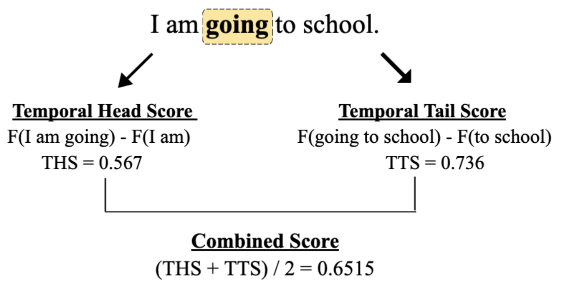


Figure 1. Figure1: Scoring function

Combined Score (CS): To capture a comprehensive understanding of each token's importance within its surrounding context, we combine THS and TTS to form the Combined Score (CS). CS leverages both scoring mechanisms, allowing us to ascertain the significance of a token based on its entire context. The calculation of CS involves taking the average of THS and TTS.

$$CS(x_i) = (THS(x_i) + TTS(x_i))/2$$

Once the importance of each token is estimated, we select the top k percentage tokens with the highest combined scores for perturbation, thereby creating an adversarial sequence.

3.1.2. White-Box Scoring:

The White-Box Scoring method is analogous to the Black-box one, in the sense that it is composed of head and tail scores. The head of the white-box scoring method computes a score for the ith token (just like its Black-box counterpart), wherein the score of the ith token is equal to the loss of the sequence of tokens up to and including the ith token minus the loss of the sequence of tokens up to and excluding the ith token. The white box tail scoring is just an analogue of the black box one, but now using the loss instead of the forward pass. The logic behind placing importance on tokens that affect the loss is the following: those tokens will have had a meaningful contribution in driving the fitted value of the input away from the ground truth. Thus, our model requires knowledge on the ground truth of the data, as well as what loss function is used to train the model (which is what makes it White-box).

3.2. Attack

Once the important words are identified we proceed to transform them, thereby generating an adversarial sample. This leads us into the discussion of what mechanism is used to transform the important words. Important words are transformed by substituting them with their synonyms based on their semantic similarity and conceptual significance.

Let $X = \{x_1, x_2, ..., x_n\}$ be a set of tokens (i.e. a given input).
At this stage all the tokens in our input have been assigned a Combined Score.

The equation for replacing x_i is the following:

$$x_i = \begin{cases} x'_i, & CS(x_i) > \lambda \\ x_i, & CS(x_i) \leq \lambda \end{cases}$$

where $CS(x_i)$ represents the combined score of x_i and λ represents the k-th percentile of scores.

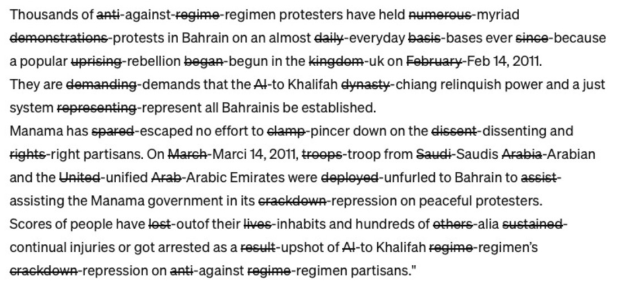
Now that we have identified which tokens will be replaced we can proceed to transform them, thereby generating an adversarial example. The question that remains is what will we replace x_i with.

x_i's replacement will be a synonym of the token's corresponding word, so as to perserve the semantics of the text. Our method for finding the synonym of a given word will be to find the word's nearest neighbor in the Counter-Fitted embedding space. This achieves our goal of finding the word's synonym, since Counter-Fitted's embedding space is designed such that words that are near to each other are synonyms.

Thus, our method for token replacement is:

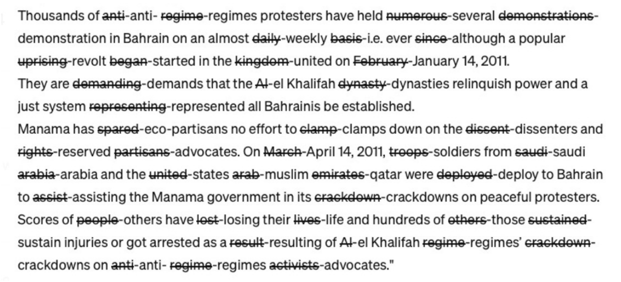
$$x_i'^{CF} = \arg \min_{x \in CF\_dictionary} \|x_i'^{CF} - x\|_2$$

Here, $\| \cdot \|_2$ represents the Euclidean distance and x_i^{CF} is x_i's corresponding vector in the Counter-Fitted embedding space.



Thousands of ~~anti~~-against-~~regime~~-regimen protesters have held ~~numerous~~-myriad ~~demonstrations~~-protests in Bahrain on an almost ~~daily~~-everyday ~~basis~~-bases ever ~~since~~-because a popular ~~uprising~~-rebellion ~~began~~-begun in the ~~kingdom~~-uk on ~~February~~-Feb 14, 2011.
They are ~~demanding~~-demands that the ~~Al~~-to Khalifah ~~dynasty~~-chiang relinquish power and a just system ~~representing~~-represent all Bahrainis be established.
Manama has ~~spared~~-escaped no effort to ~~clamp~~-pincer down on the ~~dissent~~-dissenting and ~~rights~~-right partisans. On ~~March~~-Marci 14, 2011, ~~troops~~-troop from ~~Saudi~~-Saudis ~~Arabia~~-Arabian and the ~~United~~-unified ~~Arab~~-Arabic Emirates were ~~deployed~~-unfurled to Bahrain to ~~assist~~-assisting the Manama government in its ~~crackdown~~-repression on peaceful protesters.
Scores of people have ~~lost~~-outof their ~~lives~~-inhabits and hundreds of ~~others~~-alia ~~sustained~~-continual injuries or got arrested as a ~~result~~-upshot of ~~Al~~-to Khalifah ~~regime~~-regimen's ~~crackdown~~-repression on ~~anti~~-against ~~regime~~-regimen partisans."

Figure 2. Counter Fitted Embedding word replacement



Thousands of ~~anti~~-anti- ~~regime~~-regimes protesters have held ~~numerous~~-several ~~demonstrations~~-demonstration in Bahrain on an almost ~~daily~~-weekly ~~basis~~-i.e. ever ~~since~~-although a popular ~~uprising~~-revolt ~~began~~-started in the ~~kingdom~~-united on ~~February~~-January 14, 2011.
They are ~~demanding~~-demands that the ~~Al~~-el Khalifah ~~dynasty~~-dynasties relinquish power and a just system ~~representing~~-represented all Bahrainis be established.
Manama has ~~spared~~-eco-partisans no effort to ~~clamp~~-clamps down on the ~~dissent~~-dissenters and ~~rights~~-reserved ~~partisans~~-advocates. On ~~March~~-April 14, 2011, ~~troops~~-soldiers from ~~saudi~~-saudi ~~arabia~~-arabia and the ~~united~~-states ~~arab~~-muslim ~~emirates~~-qatar were ~~deployed~~-deploy to Bahrain to ~~assist~~-assisting the Manama government in its ~~crackdown~~-crackdowns on peaceful protesters.
Scores of ~~people~~-others have ~~lost~~-losing their ~~lives~~-life and hundreds of ~~others~~-those ~~sustained~~-sustain injuries or got arrested as a ~~result~~-resulting of ~~Al~~-el Khalifah ~~regime~~-regimes' ~~crackdown~~-crackdowns on ~~anti~~-anti- ~~regime~~-regimes ~~activists~~-advocates."

Figure 3. GloVe dictionary word replacement

The figures above illustrate Counter Fitted's superiority over other embedding spaces, specifically GloVe in this example, when employed as a dictionary for word replacement. For example, in the attack that used Counter Fitted for word replacement, the word "daily" was replaced with "everyday", whereas GloVe's counterpart "daily" was replaced with "weekly". Clearly, the Counter Fitted replacement is closer in meaning to the original word than GloVe's replacement, and, again, this is due to the Counter Fitted embedding space having the property wherein words that are closer to each other share the same meaning.

4. Experiments

4.1. Tasks

In our study, we investigate adversarial examples across two natural language classification tasks. Our overarching goal is to assess the impact of adversarial perturbations on different classification models and tasks.

4.1.1. Sentiment Analysis

The sentiment analysis task involves classifying textual data from the IMDb dataset as either positive or negative. This binary classification task aims to determine the overall sentiment of 10,000 customer reviews, which can provide insights into customer opinions and attitudes toward a particular subject or service. Although the IMDb dataset contains 50,000 entries, we selected a subset of 10,000 for experimentation to expedite training and evaluation time. The IMDb dataset offers a rich source of varied text data, making it suitable for developing robust classification models.

4.1.2. Fake News Detection

The fake news detection task aims to classify text entries from a dataset containing 4,572 entries. Each entry consists of an ID, title, text, and a binary label indicating whether the news article is real (REAL) or fake (FAKE). This binary classification task is crucial in today's digital age to combat the spread of misinformation and verify the authenticity of news sources. As such, a nefarious actor will seek to spread misinformation by crafting adversarial attacks that are capable of bypassing such verifications. The dataset provides a variety of news articles spanning different topics and sources, presenting a challenging but rewarding opportunity to develop effective classification models capable of distinguishing between real and fake news.

4.2. Tokenization and Embeddings

In this study, we approach natural language processing tasks by first tokenizing the text data. Tokenization converts text into a sequence of tokens (typically words or sub-words), transforming the raw text into a format that neural network models can understand.

Following tokenization, we explore two different embedding techniques to map the tokens into dense vectors (embeddings) of uniform 300-dimensional size. Embeddings represent the semantic and syntactic information of words, allowing the models to understand relationships between words and their contexts. Moreover, an embedding space acts as a dimensionality reduction technique for the initial data, which is the size of the vocabulary.

4.2.1. Baseline Embedding

The Baseline Embedding uses a simple approach, leveraging TensorFlow's Embedding layer to directly create embeddings from the tokenizer's vocabulary. We utilized the number of unique words in our preprocessed dataset as the vocabulary size and the number of words in our longest data-point as input length. The baseline method stands in contrast to the GloVe embedding space in that the weights of the embedding require training.

4.2.2. GloVe Embedding

The GloVe Embedding technique involves using its pre-trained embeddings, which provide word vectors based on co-occurrence statistics from large text corpora. GloVe Embeddings are a type of word embedding that encode the co-occurrence probability ratio between two words as vector differences. GloVe uses a weighted least squares objective J that minimizes the difference between the dot product of the vectors of two words and the logarithm of their number of co-occurrences:

$$J = \sum_{i,j=1}^{V} f(X_{ij}) \left(w_i^T w_j + b_i + b_j - \log X_{ij} \right)^2$$

where w_i and b_i are the word vector and bias respectively of word i, $\tilde{w}_j$ and $\tilde{b}_j$ are the context word vector and bias respectively of word j, X_{ij} is the number of times word i occurs in the context of word j, and f is a weighting function that assigns lower weights to rare and frequent co-occurrences.

After loading the GloVe embeddings, we create an embedding matrix that maps each word in the tokenizer's vocabulary to its corresponding GloVe vector. The embedding layer is then initialized with this matrix and set to be untrainable, preserving the pre-trained embeddings.

4.3. Models

The Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) models were chosen for their complementary strengths in natural language classification tasks and assessing adversarial attacks. LSTM excels in modeling long-term dependencies in sequential data, which is valuable for tasks such as sentiment analysis and fake news detection, where understanding contextual relationships within sentences is crucial.

In contrast, CNNs are efficient at capturing local patterns and features in input data, making them well-suited for quickly handling large datasets. By applying convolutional layers with different kernel sizes, CNNs can identify important local features. This focus on local patterns makes CNNs more robust to minor variations in input data, which is particularly relevant for studying adversarial attacks.

Employing both models allows for a comprehensive exploration of text classification tasks and their vulnerability to adversarial attacks. LSTM prioritizes sequential relationships and long-term dependencies, while CNN emphasizes local features. By comparing the performance and vulnerabilities of these models, we can gain insights into how adversarial attacks impact natural language classification and enhance model robustness.

4.3.1. LSTM

We chose the LSTM model for its ability to effectively capture long-term dependencies in sequential data, making it well-suited for natural language processing tasks such as text classification. This model is particularly adept at handling variable-length input sequences and maintaining contextual information. The model consists of an embedding layer that transforms input text into dense vectors. These embeddings are then processed by an LSTM layer with 128 units, which captures the sequential patterns and long-term dependencies in the data. A dropout layer with a rate of 0.2 is added to prevent overfitting. The model concludes with a dense output layer with a sigmoid activation function, which is ideal for binary classification tasks. This architecture balances complexity with performance, providing efficient and accurate text classification.

4.3.2. CNN

We selected the CNN model for its efficiency in capturing local patterns and features in input data, which is valuable for text classification tasks. This model's ability to process input in parallel makes it well-suited for handling large datasets. The model starts with an embedding layer to convert input text data into dense vectors. It employs two convolutional layers with 128 filters and kernel sizes of 3 and 4, respectively, to extract local features from the input data. These convolutional layers are followed by global max-pooling layers to aggregate the output. The outputs from the pooling layers are concatenated, and a dropout layer with a rate of 0.2 is applied for regularization. The model ends with a dense output layer with a sigmoid activation function, providing an effective architecture for binary classification tasks.

4.4. Training

4.4.1. LSTM

The LSTM model was trained using two distinct embedding techniques: baseline embeddings and GloVe embeddings. The training process utilized the Adam optimizer with a learning rate of 0.001. Adam is known for its adaptive learning rate, offering efficient and effective training. Models were trained for 10 epochs without early stopping, relying on regularization through dropout to mitigate overfitting. The LSTM model with baseline embeddings achieved high accuracy on the training data, although some generalization issues were observed on the validation set due to overfitting. The LSTM model trained with GloVe embeddings showed high training accuracy with better generalization compared to baseline embeddings, yet it experienced some degree of overfitting.

4.4.2. CNN

Similarly, the CNN model was trained using both baseline embeddings and GloVe embeddings. Training employed the Adam optimizer with a learning rate of 0.001, facilitating stable weight adjustments and efficient learning. Both models underwent training for 10 epochs, utilizing a dropout rate of 0.2 to regularize the models and reduce overfitting. Although the CNN model trained with baseline embeddings initially achieved high accuracy, it began to overfit over time. The model trained with GloVe embeddings also displayed high training accuracy with better generalization compared to baseline embeddings but eventually experienced overfitting as well.

Early stopping was not applied in the initial training runs, which could have contributed to some overfitting observed in the models. Future training experiments could benefit from implementing early stopping based on validation loss to prevent overfitting and improve model robustness.

5. Numerical Results

	LSTM		CNN	
	Baseline	GloVe	Baseline	GloVe
IMDb	84.18%	83.33%	82.48%	87.07%
Fake News	91.13%	88.83%	84.87%	94.19%

Table 1. Test Accuracy with GloVe and Baseline Embedding on Different Datasets

The table presents test accuracy results of LSTM and CNN models using Baseline and GloVe embeddings on the IMDb and Fake News datasets. The LSTM models exhibit higher accuracy with Baseline Embedding, achieving 84.18% on IMDb and 91.13% on Fake News, compared to the slightly lower accuracy with GloVe. Conversely, the CNN models show superior performance with GloVe Embedding, achieving 87.07% on IMDb and an impressive 94.19% on Fake News, significantly outperforming the models using Baseline Embedding. These results suggest that the choice of embedding method plays a critical role in model performance, with Baseline Embedding being more effective for LSTM models, particularly on Fake News, while CNN models benefit substantially from GloVe Embedding. This demonstrates the importance of selecting the appropriate combination of model and embedding strategy for each dataset to maximize accuracy.

	IMDb	Fake News
LSTM-GloVe	88%	95%
CNN-GloVe	84%	86%

Table 2. Test accuracy of the models against black-box attacks

	IMDb	Fake News
Black Box	78%	60%
White Box	0%	15%

Table 3. Test accuracy of the an LSTM-Baseline model against both types of attacks attacks

6. Interpretations and Transferability

6.1. Interpretations

The tables present test accuracy results of models using GloVe Embedding and Baseline Embedding under different attack methods on two datasets (IMDb and Fake News). These results provide insights into the performance of the LSTM and CNN models when subjected to Black Box and White Box attacks.

In the first table, which reports test accuracy of LSTM and CNN models using GloVe Embedding under Black Box attack, the LSTM model achieves higher accuracy (88% on IMDb and 95% on Fake News) compared to the CNN model (84% on IMDb and 86% on Fake News). This suggests that the LSTM model may be more robust to Black Box attacks than the CNN model, particularly on the Fake News dataset. LSTM's ability to process sequential data efficiently may allow it to better recognize and retain context despite perturbations, leading to higher accuracy.

In the second table, which reports test accuracy of the LSTM model using Baseline Embedding under both Black Box and White Box attacks, we observe a significant drop in accuracy under the White Box attack (0% on IMDb and 15% on Fake News). This suggests that the White Box attack is highly effective in compromising the model's performance, potentially due to its use of internal model information to craft targeted perturbations. The higher accuracy under Black Box attacks (78% on IMDb and 60% on Fake News) indicates that these attacks are less effective, as they are crafted without direct knowledge of the model's parameters.

The differences in accuracy across models and attacks could also be attributed to the nature of the embeddings used. GloVe Embedding captures rich semantic relationships between words, providing the models with contextual understanding and robustness against attacks. In contrast, Baseline Embedding may lack the same level of sophistication in capturing semantic nuances, making the model more susceptible to attacks.

Moreover, the contrasting performances across the two datasets highlight the potential impact of dataset complexity on model robustness. Fake News data may present more challenges due to its inherent ambiguities and varied linguistic structures, which could explain the relatively lower accuracy of the CNN model and the more pronounced effect of White Box attacks.

Overall, these results emphasize the need for careful consideration of model architecture, embedding method, and attack strategy when evaluating model robustness and performance on different datasets.

6.2. Transferability

After attacking the LSTM-Baseline model (trained on the IMDB dataset) with white-box attacks, we extracted those perturbed inputs and used them to attack a CNN model trained on the same dataset. Recall that these attacks where highly effective on the LSTM model (resulting in the model having 0 percent test accuracy), and this did not change when aiming these attacks on the CNN model. In particular, the CNN model's test accuracy falling from 82.48 percent to 40. This showcases that the white-box attacks developed for the LSTM model posses the trait of transferability as they have been shown effective against a different model.

7. Conclusion

In conclusion, our findings demonstrate the considerable threat posed by White Box attacks, which exploit internal model information to create targeted perturbations. This method is highly effective in compromising model accuracy, as seen in the dramatic drop to 0% accuracy for the LSTM model on the IMDb dataset. Such attacks reveal the weaknesses of NLP models when faced with adversaries who possess deep knowledge of the model's architecture and parameters.

Conversely, Black Box attacks were less successful, resulting in more moderate declines in model accuracy. This suggests that defense mechanisms focusing on limiting the impact of White Box attacks, such as adversarial training and advanced regularization techniques, are critical for enhancing model robustness.

Moreover, the contrasting performances across the two datasets highlight the potential impact of dataset complexity and linguistic diversity on model robustness. The Fake News dataset, with its inherent ambiguities and varied linguistic structures, presents a different challenge compared to the IMDb dataset (which in comparison, is more uniform in structure), which may contribute to the varying levels of model performance observed in this study.

While embedding methods play a role in model performance, their influence there is less pronounced compared to their influence in the efficacy of attack methods, particularly White Box attacks. Overall, these findings underscore the importance of a multifaceted approach to improving NLP models' resilience to adversarial attacks, focusing on the nuances of model architecture, embedding methods, attack strategies, and dataset complexities.

7.1. Future Work

In future work, addressing the limitations of the current study and exploring new research avenues can lead to stronger and more comprehensive outcomes. The study could benefit from incorporating a broader range of neural network architectures, such as Transformer models, which offer unique strengths and could provide insights into model robustness against adversarial attacks.

Enhancing model resilience can be achieved through advanced techniques such as data augmentation, adversarial training, and regularization. Exposing models to adversarial examples during training and using various data augmentation methods can improve their ability to recognize and resist attacks.

Exploring more sophisticated attack strategies, such as linguistic manipulations and contextual attacks, could uncover new vulnerabilities in NLP models and lead to stronger defense mechanisms. Additionally, assessing the impact of these strategies across different languages and datasets can improve the generalisability of findings. Additionally, expanding the system to encompass multiclass classification scenarios presents an avenue for further exploration and enhancement.

Future research should also consider the ethical implications of adversarial attacks and their real-world applications. Developing guidelines for responsible use and evaluating the impact of deploying adversarial techniques in different contexts will contribute to more effective strategies for improving model robustness and ensuring ethical practices in natural language processing.

8. Bibliography

1 Gao, J., Lanchantin, J., Soffa, M., Qi, Y. (n.d.). Black-box Generation of Adversarial Text Sequences to Evade Deep Learning Classifiers. Retrieved April 27, 2024, from https://arxiv.org/pdf/1801.04354

2 Kuleshov, V., Thakoor, S., Lau, T., Ermon, S. (2018). Adversarial Examples for Natural Language Classification Problems. Openreview.net. https://openreview.net/forum?id=r1QZ3zbAZ

3 Nicholas Carlini and David Wagner. 2017. Towards evaluating the robustness of neural networks. In Security and Privacy (SP), 2017 IEEE Symposium on. IEEE, 39–57.

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

5 Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. 2016. Practical black-box attacks against deep learning systems using adversarial examples. arXiv preprint arXiv:1602.02697 (2016).

6 Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In Proceedings of the 25th international conference on Machine learning. ACM, 160–167

7 Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In Advances in Neural Information Processing Systems. 3079–3087.

8 Mrkšić, N., Séaghdha, D., Thomson, B., Gašić, M., Rojas-Barahona, L., Su, P.-H., Vandyke, D., Wen, T.-H., & Young, S. (n.d.). Counter-fitting Word Vectors to Linguistic Constraints. Retrieved April 27, 2024, from https://arxiv.org/pdf/1603.00892