Framework to Benchmark Adversarial Attacks on Language Models

by

Bipin Aasi

Bachelors, Toronto Metropolitan University, 2020

An MRP

presented to Toronto Metropolitan University

in partial fulfillment of the

requirements for the degree of

Masters of Engineering (M.Eng)

in the program of

The Department of Electrical, Computer, and Biomedical Engineering

Toronto, Ontario, Canada, 2023

AUTHOR'S DECLARATION FOR ELECTRONIC SUBMISSION OF A MRP

I hereby declare that I am the sole author of this MRP. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize Toronto Metropolitan University to lend this MRP to other institutions or individuals for the purpose of scholarly research.

I further authorize Toronto Metropolitan University to reproduce this MRP by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my thesis may be made electronically available to the public.

Framework to Benchmark Adversarial Attacks on Language Models

Masters of Engineering (M.Eng), 2023

Bipin Aasi

The Department of Electrical, Computer, and Biomedical Engineering

Toronto Metropolitan University

Abstract

Applications relying on Large Language Models (LLMs), such as ChatGPT, BARD, and Llamma, have gained widespread adoption across various industries, highlighting the critical need to assess their reliability and robustness in real-world scenarios [1, 2]. Evaluating LLMs' robustness often involves exposing them to adversarial datasets, where perturbations are introduced to assess their performance beyond the training data's distribution. This paper aims to enhance the methodological rigor of evaluating adversarial attacks on LLMs, focusing on fidelity and transferability metrics. Specifically, we contrast results from BERTScore, Universal Sentence Encoder (USE), and Sentence Similarity metrics, proposing improvements to benchmarking methodologies, such as Word Modification Rate rules [3]. Our experiment benchmarks three popular attacks: BERT based Adversarial Examples (BAE), Attack for Adversarial Training Recipe (A2T), and TextFooler, against BERT-based victim models, BERT, AlBERT, DistilBERT. Our contributions include sentence-level fidelity metrics for word-level attacks, an exploration

of alternative fidelity metrics, and the potential limitations of applying BERTScore to adversarial examples.

Results reveal that perturbation thresholds based on a percentage of the sentence length can lead to biases, favoring longer sentences. USE and Similarity Score metrics exhibit similar fidelity results, with BERTScore showing lower variance in values. BAE outperforms A2T and TextFooler in fidelity and transferability, with a direct correlation between fidelity and transferability. TextFooler demonstrates the highest drop in transferability between the victim model and the new model, while also having the lowest fidelity. These findings contribute to a more robust filtering mechanism for adversarial examples, ensuring semantical and meaningful preservation. Such a mechanism aids in preventing overfitting to non-naturally occurring data and reduces biases and costs associated with manual evaluation frameworks, enhancing adversarial attack benchmarking methodologies.

Acknowledgements

Put acknowledgments here.

Dedication

Put dedication here (optional).

Table of Contents

List of Tables

List of Figures

List of Appendices

Chapter 1

Introduction

Applications using Large Language Models (LLMs) such as ChatGPT, BARD or Llamma have permeated mainstream culture and are being employed across various industries, ranging from the medical field to finance, as exemplified by BloombergGPT [1]. This surge in their utilization underscores the need to ensure their reliability and robustness when exposed to out-of-sample real-world data. One effective technique for assessing their robustness is to evaluate how well the model performs against unseen adversarial datasets, which are of the nature of the training dataset. Adversarial attacks, originally pioneered in the domain of images, involve introducing perturbations to the inputs within a defined threshold, a concept that extends to the realm of natural language [8]. In an adversarial example, the perturbations should ideally preserve the original meaning of the text and induce the trained model to produce erroneous outputs [8, 7]. It has been empirically demonstrated that adversarial perturbed sentences are susceptible to losing their original meaning and coherence by the perturbations added, thereby casting doubt on the validity of the adversarial robustness of the model [3, 9]. The State-of-the-Art (SOTA) definition of an acceptable adversarial example is one that maintains semantic similarity to the original sentence. The SOTA metrics used to measure the consistency of semantic similarity are to measure the fidelity through a distance metric

and measure the transferability of the example on surrogate models [3, 10, 11, 9].

The goal of this project is to enhance the methodological rigor of previous work in order to enhance the existing evaluation metrics that facilitate cross-dataset benchmarking of top-performing adversarial attacks on language models. Specifically, for fidelity, this paper will contrast the results of BERTScore, Universal Sentence Encoder, and the Sentence Similarity metric and propose improvements to the existing benchmarking methodologies such as the Word Modification Rate rules. With these two metrics for transferability and fidelity in mind, this paper will implement an experiment to evaluate the results, benchmarking three popular attacks—BAE (BERT-Based Adversarial Examples), A2T (Attack for Adversarial Training Recipe), and TextFooler. The victim models will all belong to the family of BERT: "alberta-base-v2", "distilbert-base-uncased", "bert-base-uncased". By introducing innovative metrics in the realm of adversarial examples, this paper aims to enhance the methodological rigor of prior work of AdvGlue [3], CAT-Gen [12], Highly-transferable Adversarial Word Replacement (HAWR), [13] and PromptBench [14] this paper proposes a methodology that can benchmark language attacks regardless of the dataset.

Although notable efforts have been made to address the challenge of quantifying the quality of adversarial examples, there are opportunities for improvement. AdvGlue currently stands as the most robust methodology at the time of writing this paper. The authors of AdvGlue proposed a foundational framework that incorporates both fidelity and transferability measures, which has proven effective for GLUE. It is a hybrid framework that combines partially automatic and partially manual approaches for constructing adversarial examples tailored to the GLUE dataset. One clear weakness is its specificity to the GLUE dataset, rendering it unsuitable as a generalizable framework for ranking adversarial attacks on different datasets. Another prevailing concern lies in the rigid constraint on word-level attacks, such as TextAttack, BAE, and A2T. Our paper hypothesizes that this bias favors the removal of shorter-length sentences, and it does not assess word-

level attacks based on sentence similarity, which is a significant limitation. Additionally, while this paper adopted BERTScore to compare sentence-level similarity, it did not explore or compare other metrics before deciding to use BERTScore [3]. The authors of Controlled Adversarial Text Generation - CAT-Gen [12] focus on controlling adversarial text generation through use the of product categories as the controllable attribute. This paper uses perplexity which is shown to only work for Casual Language Models. Perplexity value is based heavily on what text the model was trained on means that perplexity scores are not comparable between models or datasets as noted by Meister and Cotterell, in [15]. In addition, this paper does not do a comprehensive job benchmarking the attacks on transferability by using only WordCNN and WordLSTM models. Similarly when evaluating "improved beam search in textual adversarial attacks", the authors do not test for a transferability metric [16]. Inversely the authors of Highly-transferable Adversarial Word Replacement - HAWR [13] measure fidelity and however do not have a test for transferability, as is expected by adversarial attacks [3]. The authors of PromptBench [14], aimed to build an open benchmark concentrating on adversarial prompts, however, they did not measure fidelity. They used a manual evaluation of attack quality centered on human acceptance [14]. This approach suffers from introducing a significant bias by favoring adversarial examples that may not necessarily uphold semantic coherence.

In this project we are making the following contributions: first, improve on existing benchmarking methodologies by implementing sentence-level fidelity metrics on word-level attacks, second, measure if there are scenarios where a word modification rule is limited, third, explore alternative fidelity metrics, comparing them to BERTScore to understand potential limitations when applying BERTScore rules to adversarial examples. The impact of a process for ranking adversarial attacks across datasets and different victim models can create a more robust filtering mechanism for adversarial examples, utilizing fidelity metrics and filter thresholds correctly. In turn, a more robust filtering

mechanism will contribute to a resilient adversarial benchmark dataset that preserves the semantics and meaning between the original and adversarial examples. A dataset of this nature will ensure that there is no severe overfitting on a dataset that does not naturally occur, as described in [17], where it is noted that "aggressive fine-tuning often causes the fine-tuned model to overfit the training data of downstream tasks and fail to generalize to unseen data." An automatic process to address the quality of adversarial attacks and their examples helps mitigate the inherent biases and costs associated with manual evaluation frameworks for assessing efficacy.

The report structure for this paper will begin with Related Work, followed by Method. The Method will contain the background required to understand the key steps taken in this project. The Method also contains the experiment design, which outlines the steps in more detail to reproduce the results. Followed by Experiment Evaluation, which defined the Experiment Objectives, software and hardware setup, Results and Discussion. The report closes with a section for Conclusion, which also contains future work. The Appendix and Bibliography are at the end of the report.

Chapter 2

Related Work

2.1 AdvGlue

The most robust methodology available for benchmarking adversarial examples for language models can be found in the paper by [3]. In their study, the authors devised a filtering process to curate a dataset that adhered to the criteria they had defined for high-quality adversarial examples within the GLUE dataset. They discovered that the majority of existing adversarial attack algorithms were prone to generating invalid or ambiguous adversarial examples, with approximately 90% of them either altering the original semantic meanings or leading human annotators astray [3]. The criteria for robust adversarial examples were divided into two key aspects: transferability, which gauges the effectiveness of examples in deceiving surrogate models, and fidelity, which assesses whether the original semantics were preserved.

The initial filtration step focused on transferability, retaining only those adversarial examples capable of successfully fooling the other two models for subsequent fidelity filtering. To filter for fidelity, the approach varied depending on whether the examples were produced by word-level attacks or sentence-level attacks. Word-level attacks select the most critical words in a sentence and perturb them using various strategies, such as

substituting words with synonyms [3, 2]. Sentence-level attacks, on the other hand, primarily manipulate the syntactic and logical structures of sentences [3, 2]. They often achieve their attack objectives by paraphrasing the sentence, altering the syntactic structures, or introducing unrelated sentences to divert the model's attention.

For measuring word-level fidelity in word-level adversarial examples, a straightforward if statement rule was applied: if more than 15% of word modifications were present between the original and adversarial examples, the example would be filtered out [3]. For assessing sentence-level fidelity, BERTScore was employed. BERTScore measures the similarity between words in candidate and reference sentences using cosine similarity in a BERT-encoded feature space. As discussed later in Section 3.1.2, this metric provides precision and recall for each word in the sentence. However, the authors did not provide specific details on the rule used to filter out sentence-level fidelity [3].

It is important to note that a word modification rate can be influenced by the sentence's length, making it easier for longer sentences to pass this test, as evidenced by the results presented in [16], particularly when compared to longer datasets like IMDB. Additionally, an important limitation, which will be further explored in the discussion section, is that adversarial examples for the Cola GLUE dataset aim to generate adversarial examples with the objective of being classified as grammatical English sentences. Therefore, if the semantic meaning remains unchanged, the transferability metric adds little value.

2.2 PromptBench

The authors in [14] generated 4,032 adversarial prompts, meticulously evaluating them across 10 tasks and 15 datasets, resulting in a total of 583,884 test samples. Their findings demonstrate that contemporary Large Language Models (LLMs) are vulnerable to adversarial prompts. Additionally, the authors provide a comprehensive analysis to

unravel the mystery surrounding prompt robustness and its transferability, utilizing the Average Performance Drop Rate (APDR) as a metric [14]. In contrast to [3], which serves as a static dataset for evaluating the adversarial robustness of input samples for the GLUE Dataset, PromptBench is positioned as an open benchmark focused on adversarial prompts. The authors observed that adversarial prompts exhibit some degree of transferability, however, results indicate that the transferability is inconsistent. Attack effectiveness is highly variable, with word-level attacks proving the most potent, leading to an average performance decline of 33% across all datasets [14].

The authors' approach to identifying adversarial prompts generated through word-level attacks, which do not preserve the semantic integrity of the original prompts, involved conducting a human-centric study. This study systematically excluded any adversarial prompts that failed to maintain their semantic consistency. The method for sampling the datasets to perform manual reviews of attack quality involved selecting 250 out of 4,000 examples for evaluation by five volunteers to determine their acceptability or unacceptability [14]. Results were considered acceptable if more than 70% of adversarial examples were viable, a sampling approach that could introduce bias. Furthermore, in the process of building an open benchmark focusing on adversarial prompts, fidelity was not assessed. The inconsistent measurements of transferability between models can be attributed to the absence of fidelity metrics measurements in their analysis.

2.3 Controlled Adversarial Text Generation - CAT-Gen

The authors of [12] aim to exert control over generated text while maintaining fluency by incorporating controllable attributes known to be irrelevant to the task and dataset type. In their attempt to attack a model for sentiment classification applied to product reviews, they employed product categories as the controllable attribute, with the intention

of preserving the sentiment of the reviews. Adversarial examples generated by CATGen exhibit notably high transferability, reflected in their low attack success rate against model re-training and architecture changes [14]. The results indicate that augmenting with CAT-Gen examples significantly enhances performance in CAT-Gen attacks, outperforming baselines that rely on narrower substitutions while maintaining a high level of accuracy in baseline attacks.

However, it's important to note that hard-coding product category words, even when considered unimportant, may still impact fidelity and, consequently, alter the underlying meaning of the text. This approach necessitates defining product categories specifically for product reviews and is inherently tied to the Natural Language Understanding (NLU) subtask and the dataset type, limiting its generalizability and applicability. Additionally, using perplexity to gauge fluency is heavily reliant on the training data used by the model, rendering perplexity scores incomparable between different models or datasets, as noted by Meister and Cotterell in [15]. Moreover, perplexity can only be calculated for causal language models, which contrasts with BERT, as discussed in Section 3.1.4, making it an unsuitable metric for this paper. Notably, the authors confined their victim models to classical convolutional text classification models (e.g., wordCNN) rather than exploring the applicability to transformer-based models.

2.4 Highly-transferable Adversarial Word Replacement - HAWR

In [13], the authors present a groundbreaking study that systematically explores the transferability of adversarial examples across text classification models. They investigate how various factors, including network architecture, tokenization scheme, word embedding, and model capacity, impact the transferability of these adversarial examples. Regardless of the attack algorithm or dataset employed, the tokenization scheme emerges as the most

influential factor affecting adversarial transferability [13]. It is followed by the network architecture, embedding type, and model capacity in terms of importance. Adversarial examples that were generated to attack the model trained with more data transfer slightly better than their counterparts [13]. To craft highly transferable text adversarial examples, the authors implement HAWR, a genetic algorithm that optimizes ensemble models based on empirical transferability across different models.

To generate adversarial examples, their algorithm imposes a requirement for a minimum percentage of words to be altered, although the specific minimum percentage is not specified. However, they set the maximum percentage of words allowed to be perturbed at 30% [13]. The experiments are conducted on two text classification datasets: Sentiment Movie Reviews and the AG News corpus (AGNEWS). For each dataset, 1,000 randomly selected test examples are subjected to attack. Instead of relying on BERT, they use an encoder-decoder model composed of LSTM, CNN, and other components [13].

It's worth noting that the evaluation metrics in their study solely focus on selecting successful adversarial examples, resulting in a relatively small denominator. This approach emphasizes transferability without incorporating metrics related to fidelity or assessing the fidelity of their examples.

2.5 Improved beam search

In [16], the authors delve into greedy-based algorithms for generating black-box adversarial examples and the limitations of their exploration within the search space. To overcome the somewhat lax constraint of semantic similarity and thoroughly explore the search space, they introduce an improved beam search method that incorporates multiple random perturbing positions. Leveraging transferable vulnerabilities observed in surrogate models, they select candidate words that are susceptible to attacks by target models. Their study demonstrates that beam search with multiple random attacking

positions outperforms the commonly used greedy search approach with word importance ranking [16]. The evaluation centered on the IMDB dataset, with target models including WordCNN, BiLSTM, BERT, and ESIM, while the considered attacks comprised PWWS, TextFooler, and PAE-R. The Universal Sentence Encoder (USE) was employed, and the performance of the attacks was assessed using three automatic metrics: robust accuracy, perturbation rate, and semantic similarity.

Empirical evidence shows that beam search with multiple random attacking positions achieves a higher success rate than the commonly used greedy search method with word importance ranking, even when operating within the same query budget. It reduces the accuracy of the WordCNN model on the IMDB dataset from 88.5% to 0.0% with a perturbation rate of only 1.42% [16]. This low perturbation rate suggests that the generated adversarial examples are challenging to detect. Conversely, higher perturbation rates were observed on the SST-2 and MNLI datasets, primarily due to the shorter texts in these datasets compared to IMDB. On average, only two words were perturbed in these datasets, indicating that their approach maintains the semantics of clean samples. Ideally, the adversarial examples generated should be transferable between similar models [3]. However the models used to measure transferability have totally different architectures and as seen in [13], the transferability of examples is highly variable depending on the tokenization technique and the architecture. This is a special case for language model as in the [8] paper, it was shown that the transferability property that some adversarial samples produced to mislead a specific model can mislead other models, even if their architectures greatly differ.

Chapter 3

Method

In order to understand benchmarking adversarial attacks for LLMs, it is critical to understand how attacks on language models are built, and the foundation of transformer-based language models.

3.1 Background

3.1.1 From Word2Vec to BERT

Language models all begin with a step to transform words into vectors that are interpretable by the models. Historically, semantic word representation pre-trained models like Word2Vec and Global Vectors for Word Representation (GloVe) laid the foundation for translating the semantic definitions of words into vector embeddings within a feature space, with a focus on breaking words down into n-grams. Word2Vec and GloVe models are trained on a corpus, resulting in similar words being clustered closely in the distance of their embeddings. These embeddings were popular when language models were composed of RNNs and CNNs [18, 19, 20].

Language models evolved from architectures like RNNs, which had a primary weakness:

the vanishing and exploding gradient problem occurring in longer sequences. Since then LSTM built on the recurrent architecture and introduced memory games to retain memory, thereby overcoming the problem [21]. The CNN architecture had a proven ability to retain the global context of the language. To solve Natural Language tasks such as translation, text summarization and other seq-to-seq tasks, language models generally took on an Encoder and Decoder architecture [22].

Additional design architectures, such as Bidirectionalism, were developed to enable models to consider context not only to their left but also to their right. Later, attention mechanisms were introduced to enhance these models, allowing them to learn word importance and weigh words accordingly. In 2017, a significant shift occurred when the authors of the "Attention is All You Need" paper introduced a streamlined network architecture that removed the dependency on RNNs, LSTMs, or CNNs [4].

In conjunction with the "Attention is All You Need" paper, the authors initially implemented a tokenization step using the well-established Byte Pair Encoding algorithm, ultimately eliminating the need for models like Word2Vec and GLOVE [4, 23, 24].

Other models such as GPT, GPT-2, RoBERTa, etc., also employ BPE. BPE strikes a balance between character- and word-level hybrid representations, making it capable of handling a large corpus.

To substitute recurrence and convolution techniques the transformer architecture introduced positional encoding was added to the input embeddings at the bottom of the encoder and decoder stacks to provide information about the relative or absolute position of tokens within the sequence. Multi-head attention allows the model to simultaneously attend to information from different representation subspaces at different positions. Consequently, it can learn to attend to different aspects of the input data, such as syntax, semantics, and positional information, all in parallel.

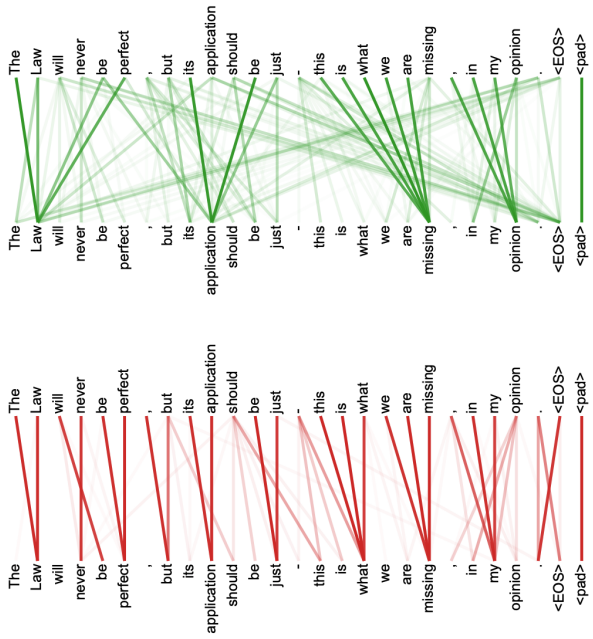

Figure 3.1: **Transformer Attention Layer** This is created by [4] to demonstrate how the attention layers on top attribute different attention values to different words of the same sentence when compared to the attention layer on the bottom. The color visualizes the magnitude of the attention.

The Bidirectional Encoder Representations from Transformer (BERT) model revolutionized and simplified the architecture, asserting that the encoder is indeed all that is required, and it introduced key features such as Bidirectionality [25]. BERT utilizes WordPiece tokenization instead of Byte Pair Encoding, as initially proposed in the transformer paper.

BERT was trained using masked language modeling (MLM) to establish a deep bidirectional representation. A portion of the input tokens was randomly masked, and the model learned to predict those masked tokens. To comprehend relationships between two sentences, that are not directly captured by language modeling, BERT leveraged techniques from Question Answering (QA) and Natural Language Inference (NLI) literature,

along with the Next Sentence Prediction (NSP) objective. It's important to note that BERT is not optimized for text generation, unlike models such as GPT, which operate causally. Instead, BERT is well-suited for embedding word semantics and serves as the foundation for BERTScore [25].

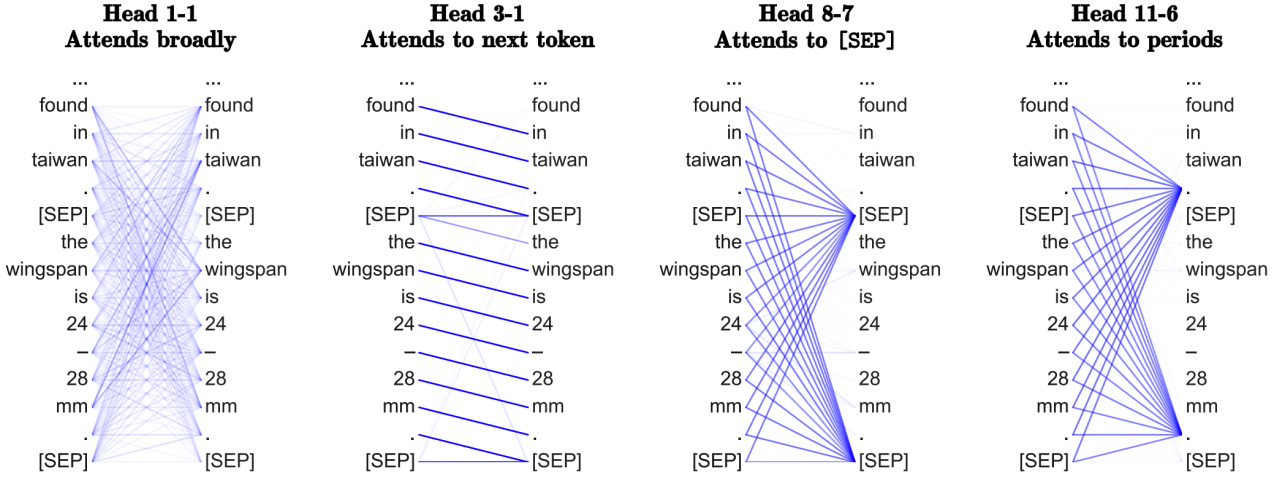

Figure 3.2: **Bert Attention Layers** The authors of [5] visualize how BERT WordPiece are attributed attention weights to. The opacity of the color represents the magnitude of the attention.

3.1.2 Semantic Similarity

BertScore

BERTScore is based on the "distilbert-base-uncased" model. It calculates a similarity score between each token in the candidate sentence and each token in the reference sentence. Leveraging the contextual embeddings of BERT, it captures the meaning of words in various contexts, enabling more precise assessments of text similarity compared to traditional word embeddings. When provided with tokenized reference and candidate sentences, BERTScore computes the cosine similarity of embeddings for each token pair,

yielding a soft measure of similarity. This metric combines precision and recall to calculate an F1 score, which serves as an evaluation criterion for the quality of generated text. To address the significance of rare words, BERTScore allows for importance weighting using Inverse Document Frequency (IDF) scores. Additionally, it incorporates baseline rescaling to enhance the interpretability of scores.

BERTScore surpasses common metrics, offering a more dependable and effective tool for model selection and evaluation across various NLP tasks [6]. While BERTScore was implemented in [3] to assess sentence-level similarity, it's worth noting that the metric may exhibit skewness when applied to longer sentences. This is because the a weighted average BERTScore is computed for each token throughout the sentence to represent similarity.

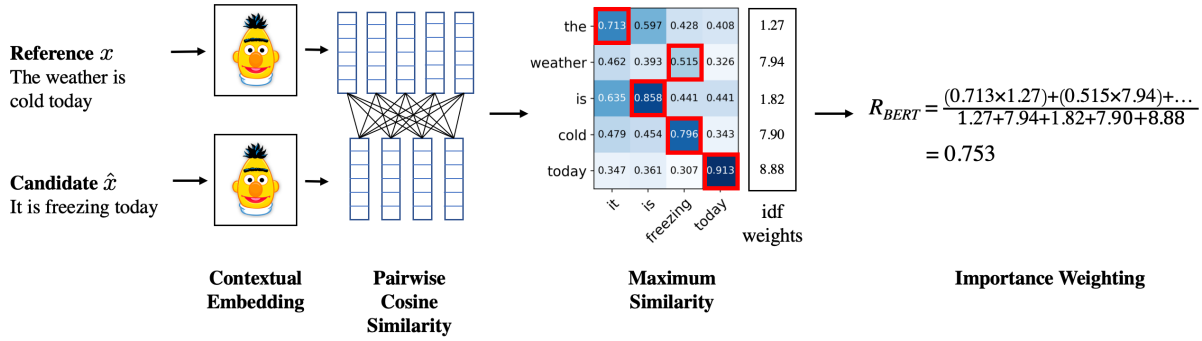

Figure 3.3: **BERTScore illustration steps** Authors of [6] show the steps which occur to produce precision, recall results for BERTScore between two sentences. For our implementation, the reference sentence is the original sentence, while the candidate is the perturbed sentence.

Universal Sentence Encoder

The Universal Sentence Encoder (USE), developed by Google in [26], is a pre-trained deep learning model that generates high-quality fixed-size vector representations, known

as embeddings. These models create sentence representations that capture semantic meanings and can be applied to various downstream tasks such as classification and similarity comparison. The paper introduces two models designed to encode sentences into embedding vectors for transfer learning in other natural language processing (NLP) tasks.

The first model employs a variant of the transformer architecture, renowned for its effectiveness in capturing contextual relationships in text. Its encoder takes a sentence as input and processes it through a sub-graph of the transformer architecture. This sub-graph utilizes attention mechanisms to compute context-aware representations of words within the sentence, considering both their ordering and identity relative to other words. While excelling in overall transfer task performance, it does consume more computing time and memory, especially with longer sentences.

The second model adopts a Deep Averaging Network architecture, where input embeddings for words and bi-grams are averaged and then fed through a deep neural network (DNN) for sentence embedding generation. Similar to the transformer encoder, this model takes a sentence as input and produces a 512-dimensional embedding. It demonstrates strong performance in text classification tasks.

USE has been employed in [16] to assess the fidelity of adversarial examples; however, this was done in isolation without accompanying transferability metrics. As discussed in [16], the USE metric was used to measure fidelity.

Sentence Similarity

The authors of [27] introduce the SentenceTransformer model, specifically "sentence-transformers/all-MiniLM-L6-v2". This pre-trained transformer network has set new state-of-the-art results in various NLP tasks, including question answering, sentence classification, and sentence-pair regression. Sentence-BERT enhances BERT/RoBERTa by incorporating a pooling operation to generate a fixed-sized sentence embedding from

multiple word-level tokens.

The pooling operation can either average the input tokens, utilize the output of the special CLS token, or default to the maximum token value. The model consists of two identical subnetworks (twin networks) that share weights. This architectural design enables efficient processing of pairs of sentences, computing embeddings for both sentences within the pair. To achieve this, the model employs pooling techniques that combine information from different parts of a sentence into a fixed-size vector, ensuring the representation captures the most relevant information for conveying the sentence's meaning.

3.1.3 Transferability

As defined in [8] transferability refers to the phenomenon where adversarial samples designed to deceive one specific model can deceive a surrogate model. An example of this is demonstrated when a local Deep Neural Network (DNN) is trained using manipulated inputs and output labels from the target victim DNN. The attacker's access is constrained to obtaining outputs for chosen inputs through an oracle mechanism. A good attack is transferable [28, 3, 13].

In [11], the authors propose a simple and effective attack method that causes aligned language models to generate objectionable behaviors. Specifically, their approach finds a suffix that, when attached to a wide range of queries for an LLM to produce objectionable content, aims to maximize the probability that the model produces an affirmative response. These attacks are highly transferable and do not attempt to maintain semantic similarity. Attacks such as these do not attempt to follow a minimum perturbation margin, as is usually done in classical adversarial attacks [9].

The adversarial transfer is not symmetric, and the transferability rates of intra-model adversarial examples are consistently higher than those of inter-model one [13]. The paper [13] proves that transferability is a function of many variables, therefore this paper

keeps these parameters constant and defined in Section 3.

3.1.4 Adversarial Examples In Language Model

Adversarial attacks were initially developed for images, involving the addition of perturbations within a specific threshold to the image while remaining imperceptible to human observers. These adversarial examples aim to preserve the image's semantics and yield the same output as the original image [8]. In the context of language models, attacks can take various forms, such as grammatical modifications, character flips, or spelling errors. Character-level attacks, which involve altering individual characters in words, are relatively easy to detect [14]. However, the majority of research has focused on word-level attacks.

Character-level attacks typically entail adversaries modifying several characters within words to create adversarial examples capable of deceiving detectors. In contrast, word-level attacks encompass a range of perturbations, including inserting, replacing, or deleting specific words through various techniques. Sentence-level attacks often involve inserting an additional sentence into the text or rewriting a sentence while preserving its original meaning. To some extent, sentence-level attacks can be viewed as a specialized form of word-level attack, achieved by adding specific words in a particular order. Multi-level attacks combine elements of all three perturbation types to achieve imperceptibility and a high success rate [2]. Figure 3.4 provides an illustration of the breakdown of different perturbation strategies.

While character-level attacks can achieve a high success rate, they are susceptible to detection due to misspellings. Adversarial training techniques can be applied to character-level attacks. In contrast, word-level modifications are often less perceptible to humans than character-level attacks, and there exists a wide variety of word-level attack strategies. However, sentence-level attacks are relatively less explored, especially in the context of text classification. It's worth noting that multi-level attacks, although

effective, are computationally more expensive and complex. For the sake of simplicity, this paper primarily focuses on word-level attacks for generating adversarial examples. However, the evaluation methodology proposed in this experiment theoretically extends to any form of attack, as it attack agnostic.

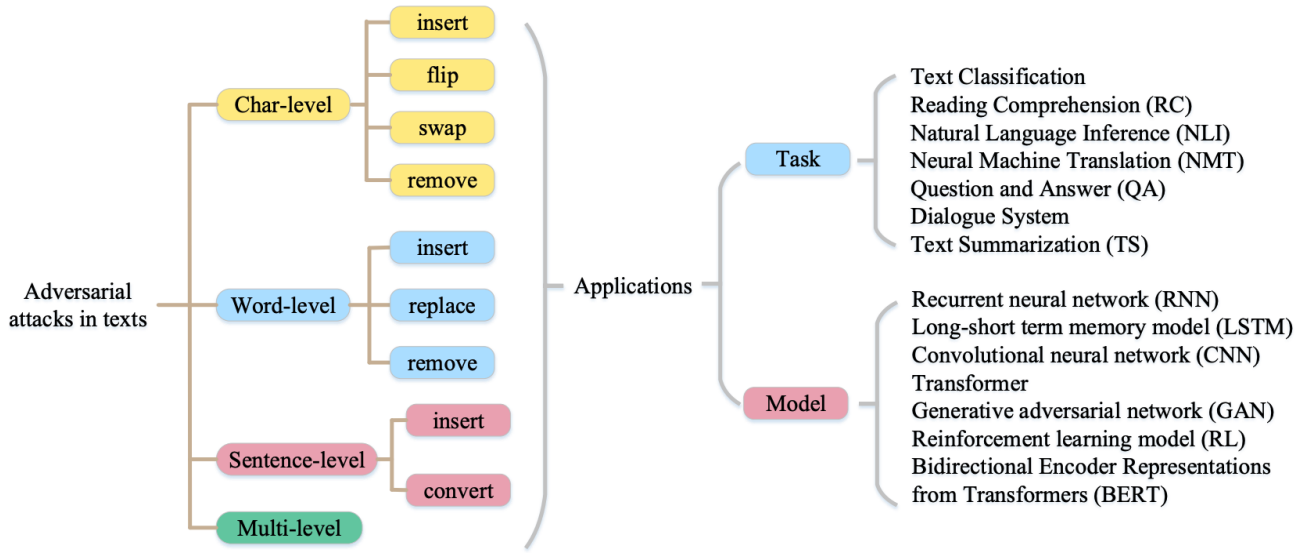

Figure 3.4: **Examples of language perturbations and Applications** Authors of [2] breakdown the types of adversarial attacks for language models. Our implementation are word level attacks with the task of Text Classification.

As outlined by TextAttack [7], adversarial attacks for language models can be divided into three steps, as illustrated in Figure 3.5. The first step involves selecting the victim language model and dataset used during inference, which will be perturbed.

The second step defines the parameters of the attack function:

- **Task-specific goal function**: This function assesses attack success based on model outputs. Examples include untargeted or targeted classifications.
- **Transformation**: Given an input, it generates a set of potential perturbations. For instance, a transformation might provide all possible synonym replacements for

a given word. Examples of transformations include word embedding word swaps, thesaurus-based word swaps, and homoglyph character substitutions.

- **Constraints**: These determine the validity of perturbations relative to the original input. For each transformed option, constraints are checked, returning a boolean value representing whether the constraint is met. Constraints may encompass factors like maximum word embedding distance, part-of-speech consistency, adherence to grammar rules, and minimum cosine similarity in sentence encodings.

- **Search Method**: This method is used to find a sequence of transformations that yield a successful adversarial example. A search consists of successive calls to obtain transformations until the search either succeeds or is exhausted. Examples of search methods include greedy approaches with word importance ranking and beam search, among others.

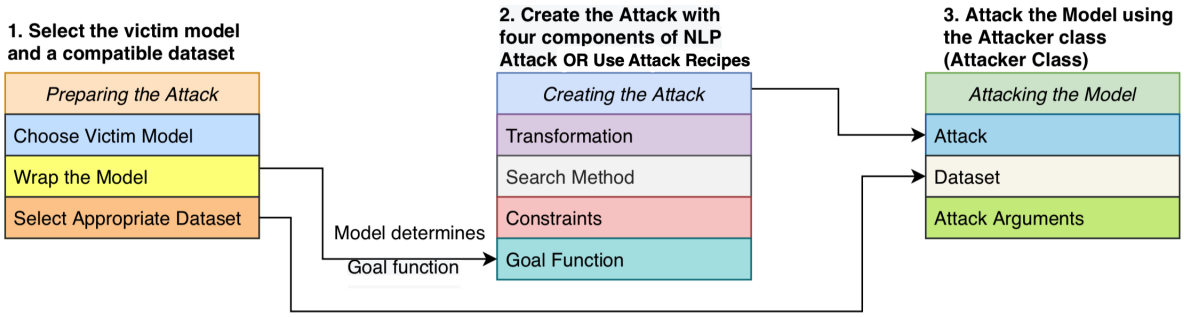

Figure 3.5: **TextAttack Steps** Authors of [7] display the key steps that were taken while producing adversarial attacks for a trained victim model and dataset to be perturbed.

The TextAttack framework classifies all language attacks based on their Goal Function, Constraints, Transformations, and Search Method. These attacks can also be grouped based on how they select the words to manipulate, including gradient-based and importance-based methods, genetic algorithms, particle swarm optimization, etc. Query-based attacks

identify vulnerable tokens by querying the output decisions and scores of the target model, applying various strategies to these tokens to generate adversarial texts [16].

While TextAttack directly implements many types of attacks, some attacks fall within the TextAttack framework without specific implementations in the TextAttack library. For example, in [11], the focus is on tricking the model by adding suffixes. Attacks with similar characteristics, meeting the requirement of transferability but not semantic similarity, will not be evaluated. The transformation strategy of adding suffixes in other current publications [29, 30]. Visual perturbations applied to text, as described in [31], currently do not exist in the TextAttack package to be ready for implementation. Visual modifications to text are often employed to obfuscate offensive comments on social media (e.g., "!d10t"). Attacks can also be built using a tree-based framework, as described by [32], which embeds discrete text data into a continuous representation space. Adversarial perturbations are optimized to achieve the target attack goal while minimizing the magnitude of the perturbation. While TextAttack does not currently have implementations for these attacks, they can be defined by specifying the "Transformation," "Constraint," and "Search Method" parameters for each attack [7].

This paper focuses on BERT-based Adversarial Examples (BAE) Attack [33], TextFooler [34], and Attacking to Training (A2T) [35]. Detailed information about these attacks is provided in Table 3.1 and the Appendix Section A. Greedy Word Importance Ranking (WIR) works by greedily selecting from a list of possible perturbations in index order, based on ranking words by importance. As it ranks words by their importance, Greedy Word Swap WIR is limited to word swap and deletion transformations [7].

Table 3.1: **Attack Details** A breakdown of the parameters which define the default BAE, TextFooler and A2T attacks in the TextAttack implementation.

	Goal Function	Constraints	Transformations	Search Method
BAE	Untargeted Classification	USE Sentence Encoding Cosine similarity	BertForMaskedLM	Greedy-WIR (Delete ranking)
TextFooler	Untargeted Classification	Word Embedding Distance, USE sentence encoding cosine similarity	WordSwapEmbedding	Greedy-WIR (Delete ranking)
A2T	Untargeted Classification	BERT	WordSwapEmbedding	Greedy-WIR (Grad ranking)

3.2 Experiment Design

The fidelity and transferability of the adversarial examples produced by each combination across three language models, three language attacks and two datasets are being measured.

When comparing the original sentence to the adversarial example across the dataset, the fidelity metrics reflect how semantically the sentences are different. High fidelity values mean sentences are very similar, while low fidelity metrics represent a large difference in semantic meaning. The relationship between two sentences and their fidelity is isomorphic, one-to-one. Ideally, when the text attack is applied, the semantics should be preserved, thereby not changing the fidelity. If the values of fidelity across the dataset do not change, it reflects that the attacks all perform the same way across sentences, and in turn, this metric does not hold much information [36]. One advantage of having a metric that has a higher distribution is that it can relative changes in semantics much easier, and it holds more information about the sentence. Our paper will compare the standard deviation of fidelity metrics across each attack, dataset, and victim model combination.

As stated earlier this paper intends to measure the following Performance of fidelity metrics on word-level attacks. Transferability is measured for each adversarial example.

The drop between the original and drop in accuracy for other models should be very similar, thereby being transferable. Sensitivity of fidelity metrics to the length of the dataset (Word modification sensitivity). The rank order of which attacks performed best. The Attack success rate can also be measured and grouped by bins of sentence lengths. We can also bin each original sentence by length, and show that there's a relationship between the percentage perturbed and the sentence length.

In order to answer these questions, the experiment includes the following steps:

1. Produce adversarial examples
    (a) Fine tune Models: Figure 3.6
        i. Process Datasets
        ii. Define model parameters and train
    (b) Implement Adversarial Attack: Figure 3.7
        - Define attack parameters
        - Attack victim model
2. Generate Fidelity metrics comparing each adversarial example to the original: Figure 3.8
3. Measure Transferability: Figure 3.9
    - Take the adversarial examples produced using a fine-tuned model victim model and an attack, and pass to other already trained fine-tuned models for inference
    - Repeat for all datasets and attack and victim model combination.
    - Calculate the Attack Success Rate across. Define the drop in accuracy from the original model, to the drop in accuracy of the other models

3.2.1 Fine Tune Models

The pre-trained models were used from HuggingFace: https://huggingface.co/bert-base-uncased, https://huggingface.co/distilbert-base-uncased, https://huggingface.co/albert-base-v2

BERT variations were chosen because we would expect the highest level of transferability [13]. A Lite BERT (ALBERT) was chosen in particular because it shares its layers across its Transformer. Therefore, all layers have the same weights. Using repeating layers results in a small memory footprint, however, the computational cost remains similar to a BERT-like architecture with the same number of hidden layers as it has to iterate through the same number of (repeating) layers [37]. DistilBERT is a transformers model, smaller and faster than BERT, which was trained on the same corpus in a self-supervised fashion, using the BERT base model as a teacher. Distillation is a compression technique in which a compact model - the student - is trained to reproduce the behavior of a larger model - the teacher - or an ensemble of models [38].

The generation of adversarial examples, as illustrated in Figure 3.5, necessitates the use of a victim model. Before subjecting the victim model to an attack, it must undergo fine-tuning with the respective dataset. The fine-tuning process aligns with the guidelines provided in the HuggingFace documentation. No specific optimizations were performed on the learning rate, and the standard values, consistent with other research papers, were utilized. Both datasets employed a "LEARNING_RATE" of 2e-05 and "NUM_EPOCHS" of 2.

The datasets are used to finetune the pre-trained language models on a specific task, IMDB Reviews and Rotten Tomatoes. Both are sentiment classification problems with binary labels. Tokenization was handled through the auto tokenize function and with the following params for rotten tomatoes: The entire train and validation set was used with a "limit_text_size" of 750, "tokenize_truncation" of 200. For the IMDB dataset, "train_size" was limited to 6000, "eval_size" limited to 800, "limit_text_size" set as 750,

and "tokenize_truncation" set as 200. The dataset was shuffled before sampling the train and eval size. The tokenization truncation and limiting text size were important preprocessing steps for IMDB specifically in order to successfully fine-tune the model, due to compute resource challenges with significantly large sentence sizes in the dataset.

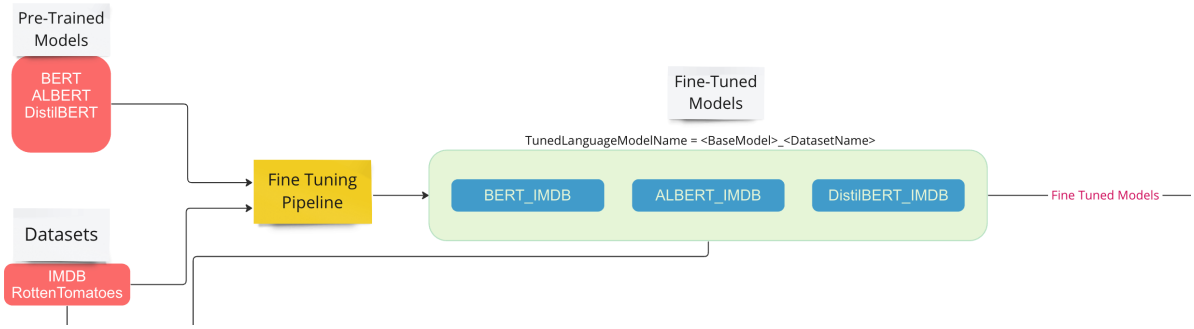

Figure 3.6: **Fine Tuning Pipeline** The process followed to convert pre-trained models from Huggingface into fine-tuned models specific to each dataset used, IMDB and Rotten Tomatoes. These fine tuned models will be used as victim models for attacks. For illustrative purposes this is showing only the fine-tuned IMDB models, however, the same process would be extended for Rotten Tomatoes

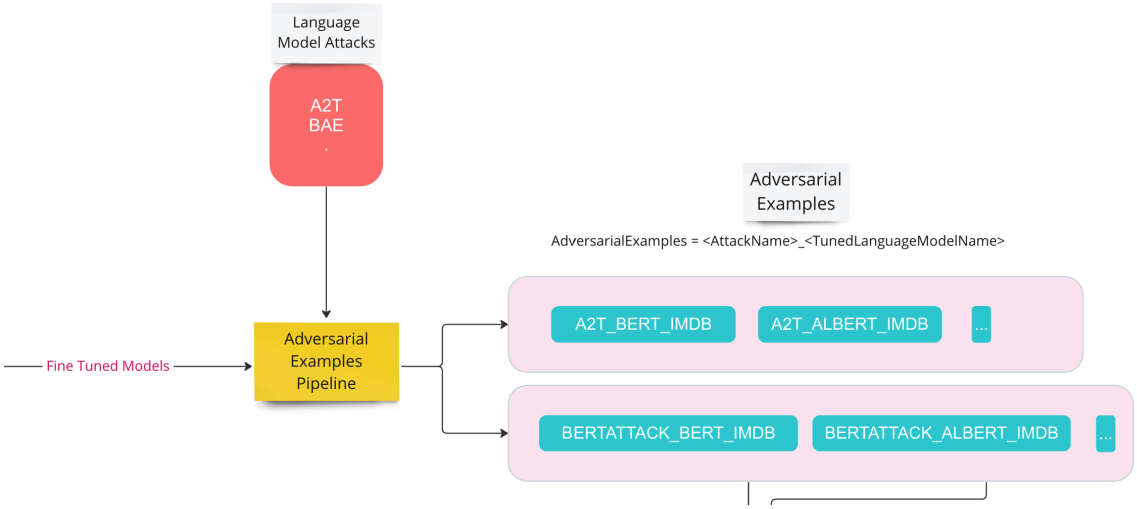

Figure 3.7: **Adversarial Examples Pipeline** The process followed to attack the victim models, from Figure 3.6, with a range of attacks to produce their respective adversarial datasets. For illustrative purposes only the A2T and BAE attacks are shown, however, the same process would extend to the TextFooler attack.

3.2.2 Implement TextAttack

As seen in Figure 3.7, the Fine Tuned Models and the Attacks Parameters are required to produce the adversarial example datasets. In this implementation, the victim model operates on the validation or test samples of the dataset. The implementation for the attack, as mentioned earlier used TextAttack and can be referenced here: https://textattack.readthedocs.io/en/latest/_modules/index.html. The evaluation split of the dataset was used to generate adversarial examples. As mentioned earlier IMDB evaluation split was limited to 800 samples and Rotten Tomatoes dataset was not limited, which had 1070 evaluation samples.

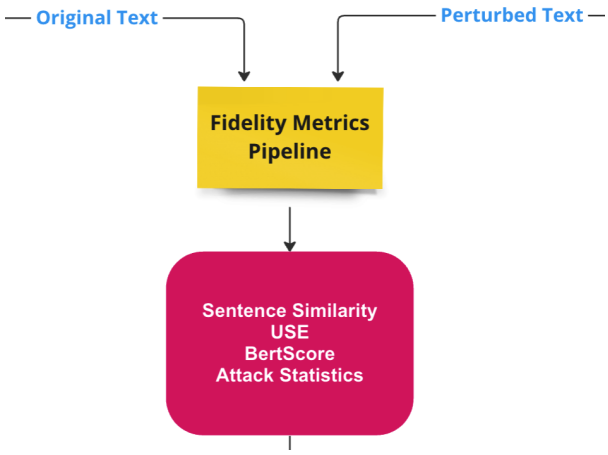

Figure 3.8: **Fidelity Method** This process takes the original text and the all possible perturbed texts generated from a combination of attacks and models as visualized in Figure 3.7.

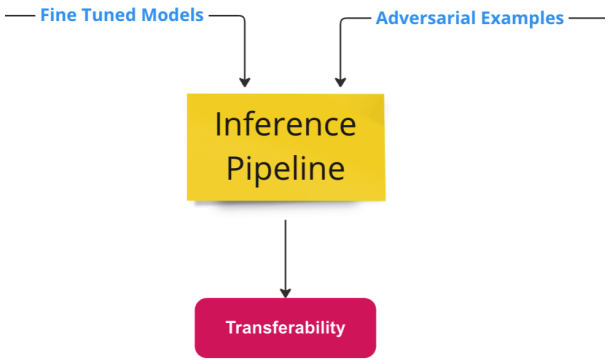

Figure 3.9: **Transferability Method** This process requires taking all fine-tuned models and producing a sentiment prediction for all adversarial examples. The change in accuracy between the origin victim model for the adversarial example, and the models the adversarial examples were not intended for are the key datapoints to calculate transferability.

Chapter 4

Experimental Evaluation

4.1 Experiment Objectives

4.1.1 Evaluating Fidelity

As mentioned earlier, because the fidelity and transferability of the adversarial examples produced by each combination across three language models, three language attacks, and two datasets are being measured, the results will be aggregated using an average. With the aggregated results we can measure the word modification rate across each dataset, specifically, we can bin each original sentence by length, and show that there's a relationship between the percentage perturbed and the sentence length. In addition, sentence-level fidelity metrics on word-level attacks will be measured by BERTScore, Universal Sentence Encoder and Sentence Similarity as described earlier.

4.1.2 Evaluating Transferability

Authors of [14] introduced a unified metric, the Performance Drop Rate (PDR),the quantify the relative performance decline following a prompt attack, offering a contextually normalized measure for comparing different attacks, datasets, and models. The PDR is

given by Transferability is measured for each adversarial example. The original and drop-in accuracy for other models should be very similar, thereby being transferable. This paper builds on that definition by calculating the average percent difference between the success rate of an attack(built on one dataset) on all victim models compared to the Inference Model. It is ideal for the value to be as close to 0 as possible.

4.2 Software and Hardware Set up

The software used for this experiment was the TextAttack framework built by [7] to implement the attacks. The workflow and pipelines as described earlier can be found here: https://github.com/bipin-a/thesis-meng/tree/main/code. A server with Tesla GPUs was used on Cuda to execute the compute.

4.3 Results

4.3.1 IMDB Dataset

Table 4.1: **IMDB Text Attack Results** Generated Statistics on the success of the attacks

Attack Name	Model Name	# of successful attacks:	Original accuracy:	Accuracy under attack:	Attack success rate:	Average perturbed word %:	# of skipped attacks:	# of failed attacks:
A2T	albert-base-v2	99	94.25	81.88	13.13	6.45	46	655
A2T	bert-base-uncased	317	92.38	52.75	42.9	5.19	61	422
A2T	distilbert-base-uncased	405	92.38	41.75	54.8	4.84	61	334
BAE	albert-base-v2	460	94.25	36.75	61.01	4.75	46	294
BAE	bert-base-uncased	424	92.38	39.38	57.37	4.31	61	315
BAE	distilbert-base-uncased	468	92.38	33.88	63.33	4.03	61	271
TextFooler	albert-base-v2	682	94.25	9	90.45	9.71	46	72
TextFooler	bert-base-uncased	632	92.38	13.38	85.52	7.92	61	107
TextFooler	distilbert-base-uncased	722	92.38	2.12	97.7	7.82	61	17

The metrics shown in Table 4.1 show that TextFooler has the highest Attack Success Rate (ASR) compared to other attacks and the highest drop in accuracy on the original model after the attack on that model. The percentage perturbations is fairly consistent within models. A2T and BAE Attacks both have the highest number of failed attacks, where sentences were perturbed, however, the model did not produce erroneous outputs.

Table 4.2: **IMDB Fidelity Results** Calculated Statistics on the Average Sentence Similarity, BERTScore, USE between the original and perturbed sentence.

Attack Name	Model Name	sentence_similarity	bertscore_result	USE_similarity
A2T	**albert-base-v2**	**0.934**	**0.97**	**0.946**
A2T	**bert-base-uncased**	**0.937**	**0.976**	**0.953**
A2T	**distilbert-base-uncased**	**0.936**	**0.977**	**0.956**
BAE	**albert-base-v2**	**0.971**	**0.979**	**0.974**
BAE	**bert-base-uncased**	**0.97**	**0.98**	**0.974**
BAE	**distilbert-base-uncased**	**0.972**	**0.981**	**0.975**
TextFooler	**albert-base-v2**	**0.905**	**0.962**	**0.919**
TextFooler	**bert-base-uncased**	**0.903**	**0.965**	**0.921**
TextFooler	**distilbert-base-uncased**	**0.927**	**0.972**	**0.944**

The metrics shown in Table 4.2, the Fidelity of the results are fairly consistent within the attack. BAE results in the highest Fidelity. BERTScore shows the highest similarity are all high compared to the other metrics, especially for TextFooler.

Table 4.3: **IMDB Transferability Results** Calculated Drop in accuracy for the original victim model, and the average drop in accuracy for the other models. A high percentage difference between these two values represents a poor transferability, as a transferable attack should have minimal dependency on which model was the victim model.

Attack Name	Model Name	% D Accuracy Victim Model	Avg % D Accuracy Other Models	Perc Diff A vs B
A2T	**albert-base-v2**	**-0.131**	**-0.043**	**-0.672**
A2T	**bert-base-uncased**	**-0.429**	**-0.144**	**-0.664**
A2T	**distilbert-base-uncased**	**-0.548**	**-0.091**	**-0.834**
BAE	**albert-base-v2**	**-0.61**	**-0.24**	**-0.607**
BAE	**bert-base-uncased**	**-0.574**	**-0.241**	**-0.58**
BAE	**distilbert-base-uncased**	**-0.633**	**-0.217**	**-0.657**
TextFooler	**albert-base-v2**	**-0.905**	**-0.217**	**-0.76**
TextFooler	**bert-base-uncased**	**-0.855**	**-0.21**	**-0.754**
TextFooler	**distilbert-base-uncased**	**-0.977**	**-0.163**	**-0.833**

The metrics shown in Table 4.3 TextFooler and A2T on average have the highest Percentage Difference between the victim model and the other models. The largest magnitude drop however goes to TextFooler, dropping from a magnitude of 0.9 to 0.2 on average. A2T drops only on average in magnitude from 0.3 to 0.1.

4.3.2 Rotten Tomatoes Dataset

Table 4.4: **Rotten Tomatoes Text Attack Results** Generated Statistics on the success of the attacks

Attack Name	Model Name	# of successful attacks:	Original accuracy:	Accuracy under attack:	ASR:	Average perturbed word %:	# of skipped attacks:	# of failed attacks:
A2T	**albert-base-v2**	**194**	**85.08**	**66.89**	**21.39**	**12.85**	**159**	**713**
A2T	**bert-base-uncased**	**266**	**85.55**	**60.6**	**29.17**	**11.68**	**154**	**646**
A2T	**distilbert-base-uncased**	**276**	**84.62**	**58.72**	**30.6**	**11**	**164**	**626**
BAE	**albert-base-v2**	**599**	**85.08**	**28.89**	**66.04**	**12.68**	**159**	**308**
BAE	**bert-base-uncased**	**580**	**85.55**	**31.14**	**63.6**	**14.55**	**154**	**332**
BAE	**distilbert-base-uncased**	**595**	**84.62**	**28.8**	**65.96**	**13.55**	**164**	**307**
TextFooler	**albert-base-v2**	**850**	**85.08**	**5.35**	**93.72**	**16.08**	**159**	**57**
TextFooler	**bert-base-uncased**	**860**	**85.55**	**4.88**	**94.3**	**17.31**	**154**	**52**
TextFooler	**distilbert-base-uncased**	**859**	**84.62**	**4.03**	**95.23**	**16.81**	**164**	**43**

The metrics shown in Table 4.4 shows once again TextFooler has the highest Attack Success Rate (ASR) compared to other attacks and the highest drop in accuracy on the original model after the attack on that model. The percentage perturbations is fairly consistent within models, i.e TextFooler will perturb on average 16.5% of words, and A2T will perturb on average 12% of words. etc. A2T and BAE Attacks both have the highest number of failed attacks, where sentences were perturbed, however the model did not produce erroneous outputs.

Table 4.5: **Rotten Tomatoes Fidelity Results** Calculated Statistics on the Average Sentence Similarity, BERTScore, USE between the original and perturbed sentence

Attack Name	Model Name	sentence_similarity	bertscore_result	USE_similarity
A2T	**albert-base-v2**	**0.905**	**0.97**	**0.892**
A2T	**bert-base-uncased**	**0.913**	**0.971**	**0.895**
A2T	**distilbert-base-uncased**	**0.919**	**0.972**	**0.899**
BAE	**albert-base-v2**	**0.909**	**0.976**	**0.907**
BAE	**bert-base-uncased**	**0.895**	**0.974**	**0.891**
BAE	**distilbert-base-uncased**	**0.905**	**0.976**	**0.902**
TextFooler	**albert-base-v2**	**0.856**	**0.96**	**0.859**
TextFooler	**bert-base-uncased**	**0.84**	**0.958**	**0.846**
TextFooler	**distilbert-base-uncased**	**0.852**	**0.96**	**0.857**

The metrics shown in Table 4.5 The Fidelity of the results are fairly consistent within the attack. BAE results in the highest Fidelity. BERTScore shows the highest similarity are all high compared to the other metrics, especially for TextFooler.

Table 4.6: **Rotten Tomatoes Transferability Results** Calculated Drop in accuracy for the original victim model, and the average drop in accuracy for the other models. A high percentage difference between these two values represents a poor transferability, as a transferable attack should have minimal dependency on which model was the victim model

Attack Name	Model Name	% Drop accuracy metrics og model	& avg drop others	% drop og_vs_others
A2T	**albert-base-v2**	**-0.214**	**-0.068**	**-0.682**
A2T	**bert-base-uncased**	**-0.292**	**-0.11**	**-0.623**
A2T	**distilbert-base-uncased**	**-0.306**	**-0.098**	**-0.68**
BAE	**albert-base-v2**	**-0.66**	**-0.358**	**-0.458**
BAE	**bert-base-uncased**	**-0.636**	**-0.406**	**-0.362**
BAE	**distilbert-base-uncased**	**-0.66**	**-0.395**	**-0.402**
TextFooler	**albert-base-v2**	**-0.937**	**-0.271**	**-0.711**
TextFooler	**bert-base-uncased**	**-0.943**	**-0.363**	**-0.615**
TextFooler	**distilbert-base-uncased**	**-0.952**	**-0.303**	**-0.682**

The metrics shown in Table 4.3 TextFooler and A2T on average have the highest Percentage Difference between the victim model and the other models. The largest magnitude drop however goes to TextFooler, dropping from a magnitude of 0.94 to 0.32 on average. A2T drops only on average in magnitude from 0.3 to 0.1.

4.4 Discussion

The results illustrate the nuanced aspects of our analysis, providing insights into threshold selection and the consequences of various metrics.

When considering the perturbation percentage as a threshold, it becomes evident that this approach is inherently influenced by sentence length. For instance, in Table 4.1, sentences from the IMDB dataset, often longer in length, exhibit an average perturbation percentage ranging from 4% to 9%. In contrast, Table 4.4 reveals that Rotten Tomatoes sentences, typically shorter, display a wider range, averaging between 12% and 17%. This discrepancy suggests that employing a uniform threshold across datasets, as proposed by [3], tends to skew samples towards longer sentences, resulting in harsher treatment of shorter ones.

Interestingly, both USE and Similarity Score exhibit remarkable similarities in the Fidelity Results Table for both IMDB and Rotten Tomatoes. However, a notable distinction emerges when analyzing the standard deviation in BERTScore metrics, which appears lower, specifically for TextFooler both IMDB and Rotten Tomatoes. This implies that BERTScore captures semantic changes with less granularity, offering less detailed information about sentences compared to USE or Sentence Similarity Metrics, as demonstrated in Appendix Fidelity Metrics for IMDB.

Notably, BAE achieves the highest fidelity, possibly attributed to its unique Transformation Method, which relies on BERT rather than conventional WordSwap techniques. In contrast, A2T and TextFooler are more reliant on WordSwap mechanisms. The Transferability Results Table further reinforces BAE's superiority, showcasing the lowest drop in transferability among attacks. Remarkably, the BAE attack consistently outperforms others in both fidelity and transferability metrics.

Conversely, TextFooler exhibits the most significant accuracy drop between the original model and surrogate models, particularly evident in both IMDB and Rotten Tomatoes

datasets, with scores of -0.78 and -0.67, respectively. It also ranks lowest in fidelity metrics. This underscores the direct correlation between fidelity and transferability.

These findings collectively argue against the implementation of a fixed word modification percentage limit, as it could unjustly eliminate potentially valuable adversarial examples. Furthermore, our results advocate for the use of sentence fidelity metrics, such as Sentence Similarity or Universal Sentence Encoder, over BERTScore. Lastly, our work underscores the effectiveness of leveraging BERT as a Transformer for perturbing words, exemplified by the BAE attack's superior performance.

Chapter 5

Conclusions

In conclusion, this study has delved into the critical domain of evaluating the robustness of Large Language Models (LLMs) when subjected to adversarial attacks. The use of LLMs in various applications necessitates thorough assessments to ensure their reliability in real-world scenarios. To this end, this paper employs innovative methodologies to enhance the benchmarking of adversarial attacks, focusing primarily on fidelity and transferability metrics.

The exploration revealed that traditional perturbation thresholds based on a percentage of sentence length can introduce biases that favor longer sentences, demonstrating the need for more robust thresholding strategies. In the assessment of fidelity, we compared BERTScore, Universal Sentence Encoder (USE), and Sentence Similarity metrics, prompting us to consider alternative fidelity metrics. This exploration elucidated potential limitations in applying BERTScore to
adversarial examples.

Furthermore, our benchmarking experiment involving three popular attacks—BERT-Based Adversarial Examples (BAE), Attack for Adversarial Training Recipe (A2T), and TextFooler against BERT-based victim models showcased intriguing results. BAE emerged as the frontrunner, excelling in both fidelity and transferability. Conversely,

TextFooler exhibited the highest drop in accuracy and the lowest fidelity, indicating a direct correlation between fidelity and transferability.

The implications of our findings extend to the development of a more resilient filtering mechanism for adversarial examples. This mechanism aims to preserve semantics and meaning between original and adversarial examples, thus guarding against overfitting on non-natural data distributions. Additionally, our proposed methodologies reduce biases and costs associated with manual evaluation frameworks, fostering advancements in adversarial attack benchmarking.

Future work includes experimenting with the tokenization of LLM and tunning the following attack variables: Search Space, Transformation, Search Function as seen in the columns of Table 3.1. In addition, as shown in [28], more queries lead to better attack performances, therefore experimenting with the maximum number of queries would also prove to be worthwhile. The transferability measurement was limited to BERT Variations, however as the original [8] suggests, transferability should also extend to models with different architecture, as such involving GPT and Falcon LLMs to this framework would validate transferability of adversarial attacks in LLM futher.

In light of these contributions, this study has expanded the methodological rigor of previous work, enhancing the evaluation metrics used for cross-dataset benchmarking of top-performing adversarial attacks on language models. As LLMs continue to play an increasingly pivotal role in real-world applications, the need for robustness assessments and innovative benchmarking methodologies becomes ever more pronounced.

Appendix A

Attack Details

TextFooler

```
Attack(
(search_method): GreedyWordSwapWIR(
(wir_method): delete
)
(goal_function): UntargetedClassification
(transformation): WordSwapEmbedding(
(max_candidates): 50
(embedding): WordEmbedding
)
(constraints):
(0): WordEmbeddingDistance(
(embedding): WordEmbedding
(min_cos_sim): 0.5
(cased): False
(include_unknown_words): True
(compare_against_original): True
)
(1): PartOfSpeech(
(tagger_type): nltk
(tagset): universal
(allow_verb_noun_swap): True
(compare_against_original): True
)
(2): UniversalSentenceEncoder(
(metric): angular
(threshold): 0.840845057
(window_size): 15
(skip_text_shorter_than_window): True
(compare_against_original): False
)
(3): RepeatModification
(4): StopwordModification
(5): InputColumnModification(
(matching_column_labels):
['premise', 'hypothesis']
(columns_to_ignore): {'premise'}
)
(is_black_box): True
)
```

BAE

```
Attack(
(search_method): GreedyWordSwapWIR(
(wir_method): delete
)
(goal_function): UntargetedClassification
(transformation): WordSwapMaskedLM(
(method): bae
(masked_lm_name): BertForMaskedLM
(max_length): 512
(max_candidates): 50
(min_confidence): 0.0
)
(constraints):
(0): PartOfSpeech(
(tagger_type): nltk
(tagset): universal
(allow_verb_noun_swap): True
(compare_against_original): True
)
(1): UniversalSentenceEncoder(
(metric): cosine
(threshold): 0.936338023
(window_size): 15
(skip_text_shorter_than_window): True
(compare_against_original): True
)
(2): RepeatModification
(3): StopwordModification
(is_black_box): True
)
```

A2T

```
Attack(
(search_method): GreedyWordSwapWIR(
(wir_method): gradient
)
(goal_function): UntargetedClassification
(transformation): WordSwapEmbedding(
(max_candidates): 20
(embedding): WordEmbedding
)
(constraints):
(0): PartOfSpeech(
(tagger_type): nltk
(tagset): universal
(allow_verb_noun_swap): False
(compare_against_original): True
)
(1): BERT(
(metric): cosine
(threshold): 0.9
(window_size): inf
(skip_text_shorter_than_window): False
(compare_against_original): True
)
(2): WordEmbeddingDistance(
(embedding): WordEmbedding
(min_cos_sim): 0.8
(cased): False
(include_unknown_words): True
(compare_against_original): True
)
(3): RepeatModification
(4): StopwordModification
(5): InputColumnModification(
(matching_column_labels):
['premise', 'hypothesis']
(columns_to_ignore): {'premise'}
)
(6): MaxModificationRate(
(max_rate): 0.1
(min_threshold): 4
)
(is_black_box): False
)
```

Table A.1: **Attack Details** TextAttack configurations for each attack used in this project.

Appendix B

Fidelitey Metrics IMDB

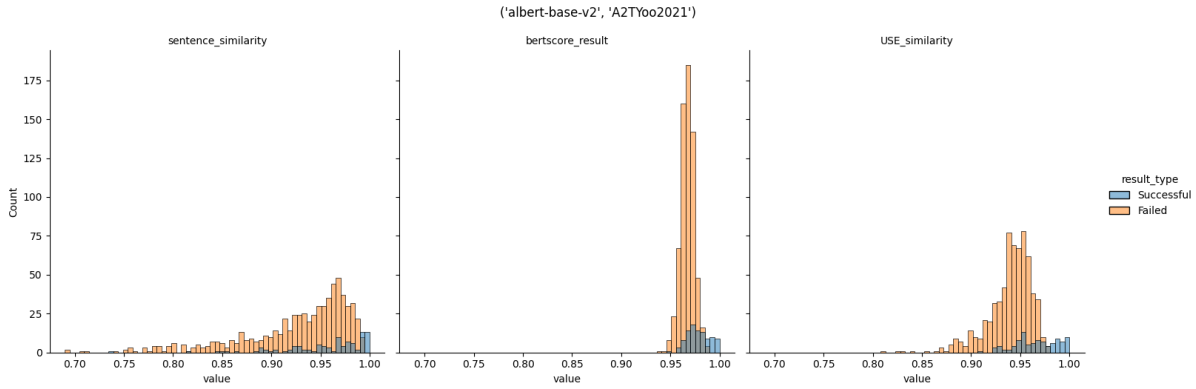

Figure B.1: albert-base-v2 A2TYoo2021

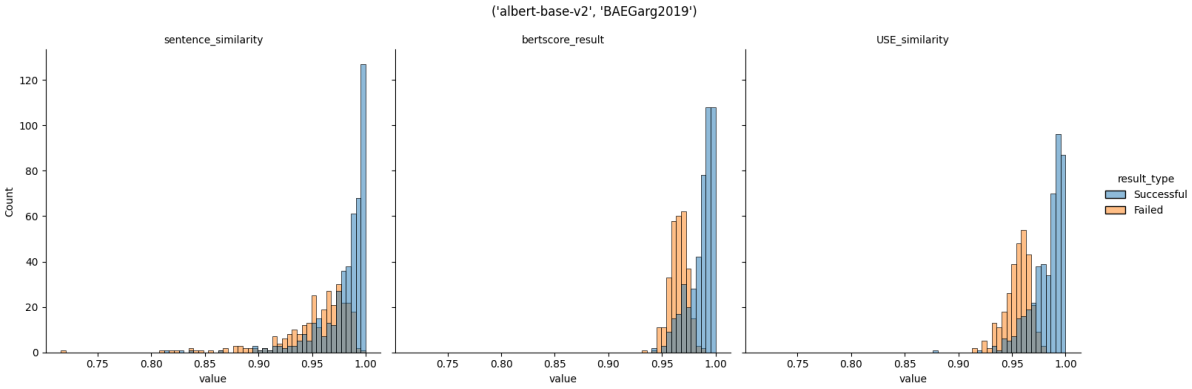

Figure B.2: albert-base-v2, BAEGarg2019

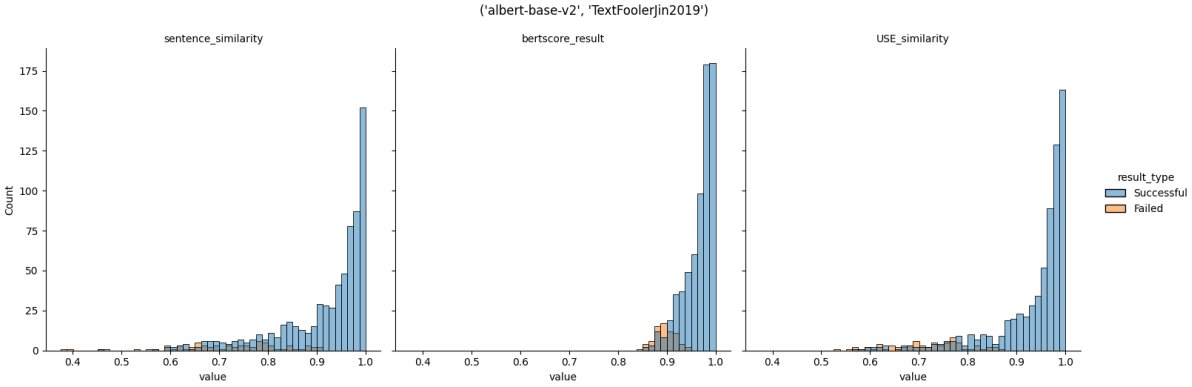

Figure B.3: albert-base-v2, TextFoolerJin2019

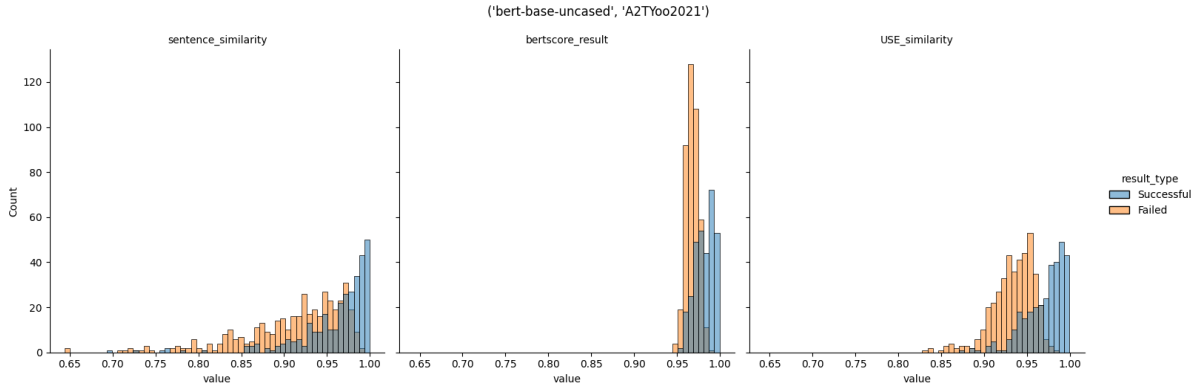

Figure B.4: bert-base-uncased, A2TYoo2021

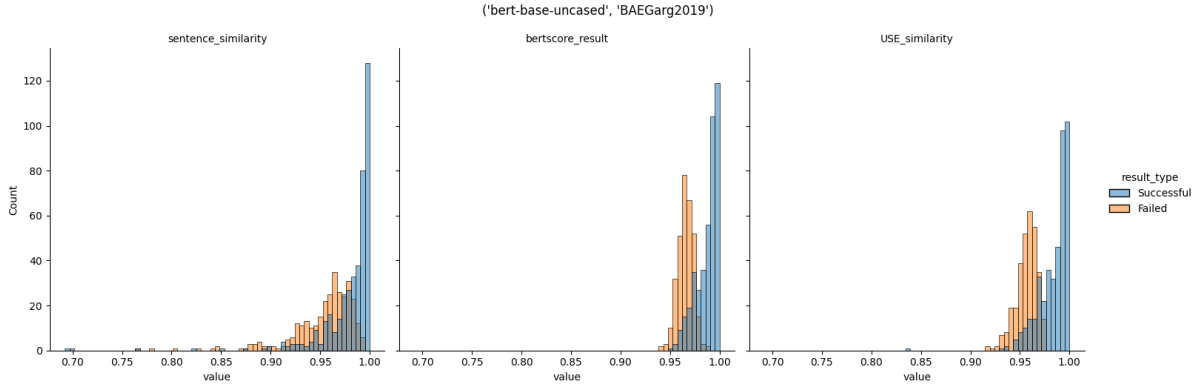

Figure B.5: bert-base-uncased, BAEGarg2019

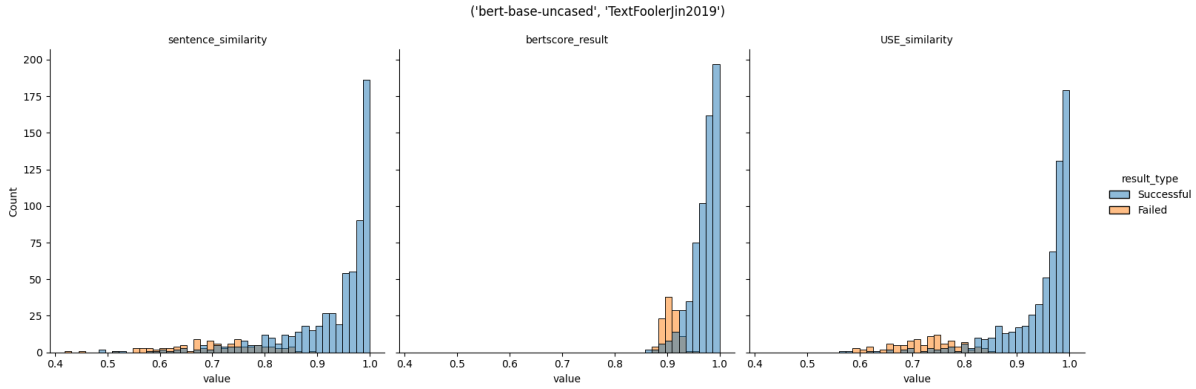

Figure B.6: bert-base-uncased, TextFoolerJin2019

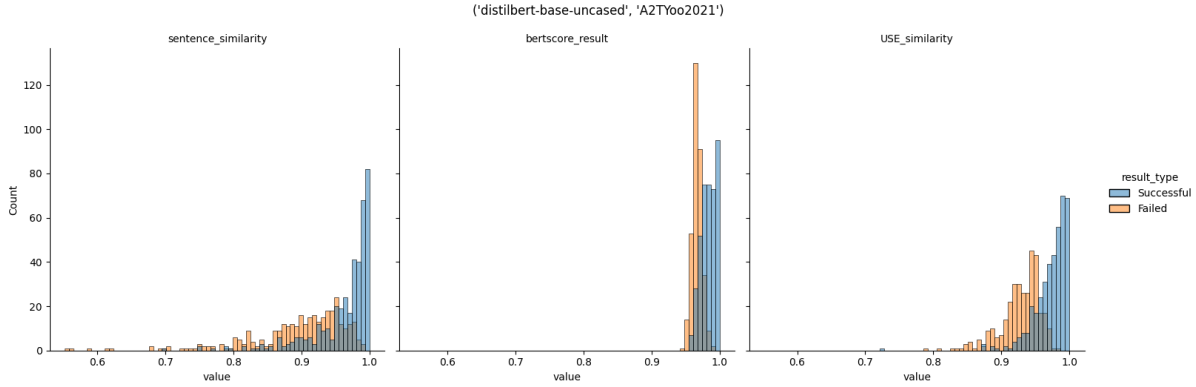

Figure B.7: distilbert-base-uncased, A2TYoo2021

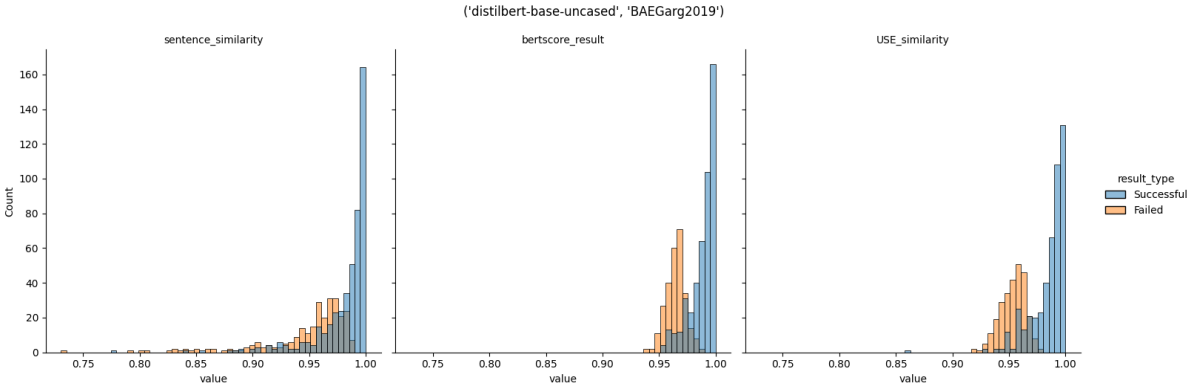

Figure B.8: distilbert-base-uncased, BAEGarg2019

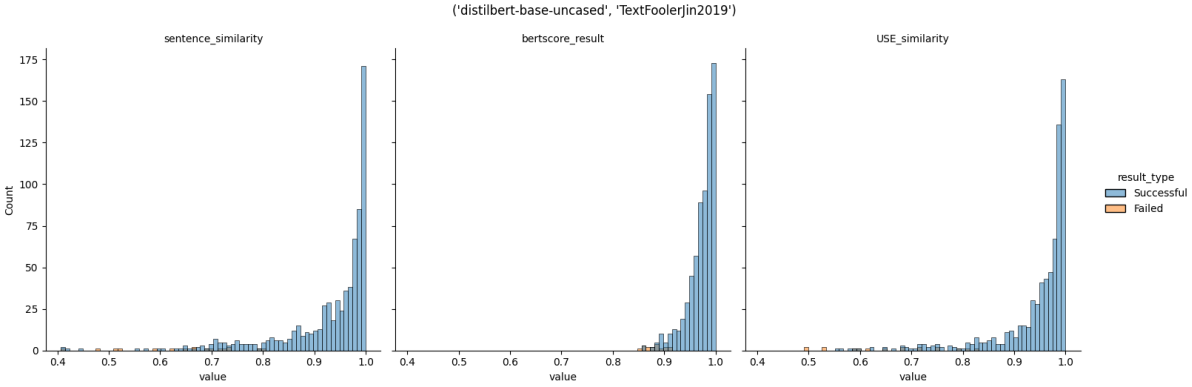

Figure B.9: distilbert-base-uncased, TextFoolerJin2019

References

[1] S. Wu, O. Irsoy, S. Lu, V. Dabravolski, M. Dredze, S. Gehrmann, P. Kambadur, D. Rosenberg, and G. Mann, "Bloomberggpt: A large language model for finance," *arXiv preprint arXiv:2303.17564*, 2023.

[2] W. Wang, R. Wang, L. Wang, Z. Wang, and A. Ye, "Towards a robust deep neural network against adversarial texts: A survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 3, pp. 3159–3179, 2023.

[3] B. Wang, C. Xu, S. Wang, Z. Gan, Y. Cheng, J. Gao, A. H. Awadallah, and B. Li, "Adversarial GLUE: A multi-task benchmark for robustness evaluation of language models," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[5] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, "What does BERT look at? an analysis of BERT's attention," in *Proceedings of the 2019 ACL Workshop*

BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, (Florence, Italy), pp. 276–286, Association for Computational Linguistics, Aug. 2019.

[6] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "Bertscore: Evaluating text generation with bert," in *International Conference on Learning Representations*, 2020.

[7] J. Morris, E. Lifland, J. Y. Yoo, J. Grigsby, D. Jin, and Y. Qi, "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*, pp. 119–126, 2020.

[8] N. Papernot, P. McDaniel, and I. Goodfellow, "Transferability in machine learning: from phenomena to black-box attacks using adversarial samples," 2016.

[9] W. E. Zhang, Q. Z. Sheng, A. Alhazmi, and C. Li, "Adversarial attacks on deep-learning models in natural language processing: A survey," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 11, no. 3, pp. 1–41, 2020.

[10] X. Liu, H. Cheng, P. He, W. Chen, Y. Wang, H. Poon, and J. Gao, "Adversarial training for large neural language models," *arXiv preprint arXiv:2004.08994*, 2020.

[11] A. Zou, Z. Wang, J. Z. Kolter, and M. Fredrikson, "Universal and transferable adversarial attacks on aligned language models," 2023.

[12] T. Wang, X. Wang, Y. Qin, B. Packer, K. Li, J. Chen, A. Beutel, and E. Chi, "Cat-gen: Improving robustness in nlp models via controlled adversarial text generation," *arXiv preprint arXiv:2010.02338*, 2020.

[13] L. Yuan, X. Zheng, Y. Zhou, C.-J. Hsieh, and K.-W. Chang, "On the transferability of adversarial attacksagainst neural text classifier," *arXiv preprint arXiv:2011.08558*, 2020.

[14] K. Zhu, J. Wang, J. Zhou, Z. Wang, H. Chen, Y. Wang, L. Yang, W. Ye, N. Z. Gong, Y. Zhang, and X. Xie, "Promptbench: Towards evaluating the robustness of large language models on adversarial prompts," 2023.

[15] C. Meister and R. Cotterell, "Language model evaluation beyond perplexity," 2021.

[16] B. Zhu, Z. Gu, Y. Qian, F. Lau, and Z. Tian, "Leveraging transferability and improved beam search in textual adversarial attacks," *Neurocomputing*, vol. 500, pp. 135–142, 2022.

[17] H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and T. Zhao, "SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, 2020.

[18] J. Pennington, R. Socher, and C. Manning, "GloVe: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Doha, Qatar), pp. 1532–1543, Association for Computational Linguistics, Oct. 2014.

[19] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems* (C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, eds.), vol. 26, Curran Associates, Inc., 2013.

[20] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2013.

[21] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, p. 1735–1780, nov 1997.

[22] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," 2014.

[23] P. Gage, "A new algorithm for data compression," *C Users J.*, vol. 12, p. 23–38, feb 1994.

[24] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Berlin, Germany), pp. 1715–1725, Association for Computational Linguistics, Aug. 2016.

[25] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[26] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. St. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, B. Strope, and R. Kurzweil, "Universal sentence encoder for English," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, (Brussels, Belgium), pp. 169–174, Association for Computational Linguistics, Nov. 2018.

[27] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," 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)*, pp. 3982–3992, Association for Computational Linguistics, Nov. 2019.

[28] X. He, L. Lyu, Q. Xu, and L. Sun, "Model extraction and adversarial transferability, your bert is vulnerable!," *arXiv preprint arXiv:2103.10013*, 2021.

[29] E. Wallace, S. Feng, N. Kandpal, M. Gardner, and S. Singh, "Universal adversarial triggers for attacking and analyzing nlp," *arXiv preprint arXiv:1908.07125*, 2019.

[30] F. Perez and I. Ribeiro, "Ignore previous prompt: Attack techniques for language models," *arXiv preprint arXiv:2211.09527*, 2022.

[31] S. Eger, G. G. Şahin, A. Rücklé, J.-U. Lee, C. Schulz, M. Mesgar, K. Swarnkar, E. Simpson, and I. Gurevych, "Text processing like humans do: Visually attacking and shielding nlp systems," *arXiv preprint arXiv:1903.11508*, 2019.

[32] B. Wang, H. Pei, B. Pan, Q. Chen, S. Wang, and B. Li, "T3: Tree-autoencoder constrained adversarial text generation for targeted attack," *arXiv preprint arXiv:1912.10375*, 2019.

[33] S. Garg and G. Ramakrishnan, "BAE: BERT-based adversarial examples for text classification," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Online), pp. 6174–6181, Association for Computational Linguistics, Nov. 2020.

[34] D. Jin, Z. Jin, J. T. Zhou, and P. Szolovits, "Is bert really robust? a strong baseline for natural language attack on text classification and entailment," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, pp. 8018–8025, 2020.

[35] J. Y. Yoo and Y. Qi, "Towards improving adversarial training of nlp models," 2021.

[36] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

[37] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," in *International Conference on Learning Representations*, 2020.

[38] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," *ArXiv*, vol. abs/1910.01108, 2019.