RoBERT: Robust BERT

Anonymous ACL submission

Abstract

Sentiment classification is a common and fundamental task in the field of natural language processing. The process involves analyzing and classifying pieces of text to label the sentiment expressed or contained in it, examples being positive or negative. In this paper, we attempt to both attack and create robustness in BERT-base, a pretrained variant of BERT (Devlin et al., 2019) (Bidirectional Encoder Representations from Transformers) , which is a state-of-the-art language model developed by Google commonly used for the task of sentiment classification. We attempt to demonstrate weaknesses in BERT for sentiment classification tasks by developing both gradient-based and algorithm-based attacks and perturbing the sentiment data set to create adversarial examples. The second premise of this paper is to develop a framework for creating robustness in language models against such attacks using adversarial training. Our experiments show significant drops in performance after creating adversarial perturbations to the test set. We also observe significant increase in performance against attacks in RoBERT after adversarial training which highlights the effectiveness of our framework for creating robustness.

1 Introduction

Sentiment classification is an important task in the field of natural language processing and has various applications such social media monitoring, customer feedback analysis, market research, financial trading, and many other commercial and research applications. With the advancement of AI research and the breadth of applications of deep learning ever increasing, adversarial attacks are becoming more relevant concerns as more processes in our workplace and every lives become automated by AI. Vulnerabilities in large language models can create problems such as exacerbating social and cultural biases (Naous et al., 2023), exposing private information (Li et al., 2023), as well as concept drift. These weaknesses raises risks in deployment in safety-critical applications such as autonomous vehicles, healthcare, or finance. Integrating robustness into language model is key to mitigating the potential damage of unreliable and unsafe behaviors of language models when facing adversarial manipulation.

In this paper, we focus on generating adversarial attacks against the state-of-the-art pretrained model BERT-base. BERT is the most widely adopted for the task of sentiment analysis and its bidirectional architecture allows the understanding of context in which words appear in a sentence, which captures nuanced relationships between tokens and allows it to grasp the sentiment expressed. BERT-base is a state of the art Transformer that has an encoder-only architecture consisting of 12 encoder stacks with a hidden dimension of size 768 and 12 attention heads per layer. We choose BERT as a strong baseline for solving typical sentiment classification problems and then develop adversarial attacks as an attempt to break BERT's performance. To achieve this, we attempted both heuristic-based letter-level substitutions to replicate common typos based on keyboard distance as well as gradient-based attacks which we follow the framework of Projected Gradient Descent. (Madry et al., 2019)

The rest of our experiment is focused on using the adversarial attacks to generate adversarial examples to create a new data set. We attempt to show that using the adversarially perturbed examples to train BERT will create robustness and allow it to perform better against both gradient-based and algorithmic-based attacks. We call our proposed model RoBERT, designed to be a robust variant of BERT-base trained using adversarial examples. Our goal is to not only show the performance of RoBERT against gradient-based attacks and perturbations, but as well as establishing a framework for adding robustness to language models.

The organization of the remaining sections of

this paper is as follows: The related works address the inspirations used for our experiments as well as competing solutions to the task of creating robustness in language models. Next, in the Proposed Model section we will discuss the architecture of BERT-base as well as the hyperparameters and specific settings used for training. Subseqeuently, in our Experiment Details section we will discuss the datasets and methods used for generating attacks and the foundations behind gradient based attacks (in our case Projected Gradient Descent) such as FGSM (Fast Gradient Sign Method) and the saddle point problem. Lastly, in our Results section we will discuss the performances of our models on both original and adversarial examples and show the affects of perturbations at both training and inference time. Our findings indicate that our framework for developing gradient based attacks is not only effective in exposing the vulnerabilities of BERT, but also preserves semantics and ground-truth while providing an effective perturbed data set for adversarial training and creating robustness which is manifested in our model RoBERT.

2 Related Work

Analogous to most other NLP tasks, recent text classification research has heavily relied on Transformer-based models (Vaswani et al., 2017). This recent research trend follows the increasing usages of Transformers as many previous algorithmic and deep learning problems found benefits in integrating Transformers and attention mechanisms. As a result, language models such as BERT (Devlin et al., 2019), XLNet (Yang et al., 2020) GPTx (Yenduri et al., 2023) have been extensively used in text classification research. Gradient-based attacks have in fact been present in modern deep learning research, however the genesis of such approaches stem from image classification tasks primarily involves Convolutional Neural Networks rather than Transformers and attention-based models. Fast Gradient Sign Method is an example of simple and effective approach involving adding a single small gradient based perturbation to a sample in the direction that increases loss. (Goodfellow et al., 2015). Projected Gradient Descent was developed a few years after this, creating a more powerful and multi-step method of generating adversarially perturbed samples. (Madry et al., 2019) However, while these methods have proven to be effective attacks, they have primarily been experimented on image classification datasets such as MNIST and there is significantly less research on the effectiveness of gradient based attacks on language models. White-box attacks like letter-level typos and word or synonym substitution is commonly explored while black-box attacks are not due to the difficulty in generating appropriate attacks while preventing breakage of sentence semantics and the original ground-truth of the classification label. There has been some research into black-box gradient based attacks on Transformers in recent years (Guo et al., 2021) and even using Transformers to generate the adversarial examples (Li et al., 2020). While these experiments demonstrate effective attacks and methods of generating perturbations, they have not explored adversarial training to create robustness in language models which is what we are attempting to demonstrate in this paper. Gradient-based perturbations on textual samples are difficult to quantitatively determine whether the perturbations are faithful to the original sentences semantics and ground truth label. Cosine similarity measurement and regularization of gradients are common methods of measuring the degree of perturbation and limiting excess changes. We utilize such methods in our own experiments but unlike other research experiments, we attempt to also experiment with adversarial training using the perturbed examples to not only create a framework for generating robustness in language models but also to prove the validity of the adversarial examples generated as faithful and trainable data sample themselves. Noisy or defunct adversarial samples generated by poorly controlled gradient-based attacks will lead to language models training on false positives or false negatives and could potentially decimate performance on the original data samples. (Boucher et al., 2021) By measuring and comparing the performance our adversarially trained model RoBERT against BERT-base on both original and adversarial data sets, we confirm the validity of our regularized perturbations and the robustness of RoBERT and its ability to generalize to new unseen attacks.

3 Proposed Model

Our model, RoBERT, is fine-tuned on sentiment dataset containing orignal tweets as well as adversarially perturbed examples mixed in. Our model uses transfer learning from BERT-base-uncased which is we imported from the huggingface library Transformers, which provides tools and APIs for

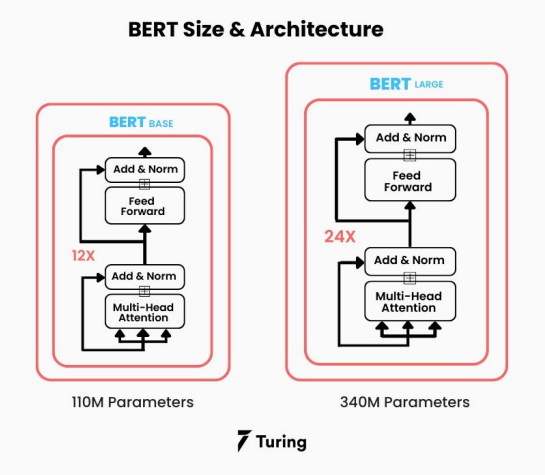

Figure 1: BERT-base model architecture vs BERT-large

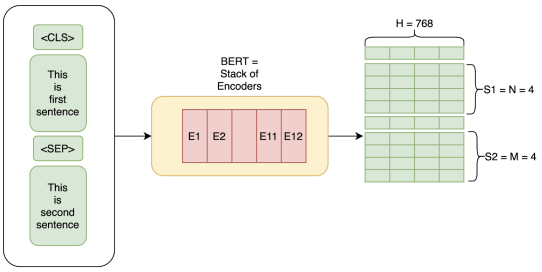

Figure 2: BERT encoding

downloading language models loaded and trained using PyTorch. BERT-base is a encoder-only multi-layer bidirectional Transformer like other BERT variants. BERT-base contains 12 encoder-stacks with 12 attention heads in each layer and a hidden dimension of 768 unlike its larger variant BERT-large which contains 24 encoder stacks with a hidden dimension of 1024 and 16 attention heads per layer. For computation efficiency and common benchmarks, we opted to use the smaller model as a baseline for text classification as opposed to its larger variant.

We elaborate on the details of the network's inputs as well as its outputs and loss function to better illustrate the learning process. The Transformer network BERT-base first uses an encoder only architecture to extract features from the text inputs tokenized and preprocessed using BertTokenizer (also imported from huggingface). The model contains a vocabulary embedding layer, segment embedding layer, and as well as a positional encoding layer which transforms the input tokens into vector representations encoding not only semantic meaning but as well as the sequential features, groupings of sub-phrases in a multi-sentence text, as well as the relevance of each individual token in the sentence. These features are captured and processed through the encoder stacks which consists of alternating multi-head self attention layers as well as a position-wise fully connected feed forward network. And finally fed into a classification head which consist of one linear layer used for binary classification since our labels only consists of positive or negative samples.

We choose to use cross-entropy as the loss function for classification. In binary classification we use this cross entropy loss function:

$$L^* = -(y \log(p) + (1 - y) \log(1 - p)) \quad (1)$$

where y is the correctness (1 = True, 0 = False) of the predicted class and p is the predicted probability of that class. This loss functions allows the model to learn a mapping between the input sentences and their respective sentiment classes. (0 = Negative and 1 = Positive)

Our transformer model aims to minimize this loss and find the model parametrs that result in the MLE or maximum likehood estimation of the of the true sentiment classification label when given a sentence to classify.

4 Experiments

We develop a heuristic-based letter-level substitution attack as a white-box baseline for attempting to disrupt BERT's performance in sentiment classification.

This first approach to confusing the BERT model drew inspiration from typos, short for typographical errors. Online social media is laden with typos, so we felt that this was representative of potential real world data. We drew inspiration from the fact that most typos occur from mistrokes on the keyboard. We accordingly derive typos by occasionally replacing a letter in a Tweet with a letter nearby on the keyboard. By doing this, we aim to make our BERT robust against typos that may occur in the data and maintain its ability to correctly classify sentiment when there are typographical errors.

For each key on the keyboard, we generated a dictionary of nearby letters that represent potential key-substitutions. Then, at a frequency of 15%, we substituted a given character for a character nearby on the keyboard. As an example, we have an original Tweet, *"I dont like deep learning,"* which has negative sentiment and BERT

classifies it as such. The letter 'J' is just above the letter 'N', on the QWERTY keyboard, so it is a realistic substitution that yields us *"I dojt like deep learning,"* which we, as humans, would still interpret as having negative sentiment. However, after we introduce the typo, BERT classifies it having positive sentiment. Our experimentation here is to reduce the odds that this happens.

We then attempt gradient-based attacks, and we decided to use Projected Gradient Descent to launch black box attacks that aim to maximize loss for BERT. Projected Gradient Descent is the more powerful version of Fast Sign Gradient Method which was introduced first as a simple but effective way of perturbing images from the MNIST dataset to create adversarial examples. (Goodfellow et al., 2015). We describe the algorithm of Fast Gradient Sign Method below.

Fast Gradient Sign Method (FGSM):

Given a model $f(\mathbf{x};\theta)$ parameterized by θ, and a loss function $\mathcal{L}$, the FGSM generates an adversarial example $\mathbf{x}'$ by perturbing the input $\mathbf{x}$ in the direction of the gradient of the loss with respect to the input.

1. **Input:** Original example $\mathbf{x}$, model parameters θ, loss function $\mathcal{L}$, perturbation magnitude ϵ.
2. **Compute gradient:**

$$\nabla_{\mathbf{x}} = \frac{\partial \mathcal{L}(f(\mathbf{x};\theta), y)}{\partial \mathbf{x}}$$

3. **Generate adversarial example:**

$$\mathbf{x}' = \mathbf{x} + \epsilon \cdot \text{sign}(\nabla_{\mathbf{x}})$$

where $\text{sign}(\cdot)$ returns the sign of each element.

While the above method is effective, modern research suggests that Project Gradient Descent is a more powerful multi-step version that generates more effective attacks. We use this attack to perturb inputs and control the gradients by clipping them with small epsilon values between 0.1 and 0.15. We describe the algorithm below.

Projected Gradient Descent (PGD):

Given a model $f(\mathbf{x};\theta)$ parameterized by θ and a loss function $\mathcal{L}$, the objective is to find an adversarial example $\mathbf{x}'$ close to the original example $\mathbf{x}$ with minimal perturbation while causing a misclassification.

1. **Initialize:** Start with a copy of the original example $\mathbf{x}_0$.
2. **Iterate until the sentence changes or reaches a certain number of iterations**
    1. Compute the gradient of the loss: $\nabla_{\mathbf{x}} \mathcal{L}(f(\mathbf{x};\theta), y)$
    2. Update the adversarial example: $\mathbf{x}_{t+1} = \mathbf{x}_t + \alpha \cdot \text{sign}(\nabla_{\mathbf{x}} \mathcal{L}(f(\mathbf{x}_t;\theta), y))$

    where α is the step size, and although not notated here, we do a element-wise clipping operation on the gradients to ensure that the perturbation is within a specified ϵ-norm ball around the original example.

While the above algorithms serve as approaches to develop adversarial attacks against deep learning models, the second part of our experiment involves developing a theoretically grounded and reproducible framework for creating robustness against adversarial attacks and we form our experiments around the Saddle Point Problem. This concept is central to the second part of our study as it is a theoretical framework for adversarial training, involving an inner maximization problem by the attacker (Projected Gradient Descent) and then an outer minimization problem by our model (RoBERT). The algorithm for the saddle point problem can be described below.

Saddle Point Problem:

Consider a machine learning model parameterized by θ with a loss function $\mathcal{L}(\theta, x, y)$, where x is the input data and y is the ground truth label. Adversarial training involves finding model parameters that simultaneously minimize the loss for the defender (the model) and maximize the loss for the attacker (the adversary). This can be formulated as a min-max optimization problem:

$$\min_{\theta} \max_{\delta} \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\mathcal{L}(\theta, x, y) - \lambda \cdot \mathcal{L}(\theta + \delta, x, y)\right]$$

where:

- θ are the model parameters to be optimized.
- δ represents the adversarial perturbation.
- $\mathcal{D}$ is the data distribution.
- $\mathcal{L}(\theta, x, y)$ is the standard loss for the defender.
- $\mathcal{L}(\theta + \delta, x, y)$ is the adversarial loss for the attacker.
- λ is a trade-off parameter that controls the strength of the adversarial perturbation.

The goal is to find θ^* such that:

$$\theta^* = \arg\min_{\theta} \max_{\delta} \mathbb{E}_{(x,y)\sim\mathcal{D}} [\mathcal{L}(\theta, x, y) - \lambda \cdot \mathcal{L}(\theta + \delta, x, y)]$$

Using these concepts we conduct our experiment in the following order:

First we train BERT on the Sentiment140 dataset to achieve a baseline accuracy on the original test set. We used a batch size of 16, AdamW optimizer, and a learning rate of 2e-5 (0.00002). This was able to achieve good accuracy on the test set and gives us a baseline performance to develop attacks against. Next, we find the optimal parameters for projected gradient descent by minimizing BERT's accuracy at inference time against adversarial samples as well as manually evaluating the semantics and ground-truth of the perturbed inputs to control the degree of perturbation and optimize the epsilon clipping parameters. We decided that using an alpha (perturbation rate) of 1, epsilon (gradient clipping threshold) value of 0.1, and iterations of 20 serve as good parameters that generate adversarial results while preserving semantics. Because one of our goals is to preserve the faithfulness and meaning of sentences, we opted to mitigate the degree of perturbations and control them to where the attacks are just enough to cause a portion of the perturbed inputs to misclassify while limiting semantic or ground-truth destruction as much as possible.

Our next experiment consists of using the perturbed adversarial dataset to train BERT in an attempt to create a more robust version (RoBERT). The perturbations are used to create the adversarial training set of 10000 perturbed samples, which we mixed with 10000 of the original samples in order to preserve accuracy on the original population of tweets. We run each of the models (original and adversarial trained) against each of the data sets (original and adversarial) which results in a total of 3x3 which is 9 experiments.

The adversarial data sets for the Projected Gradient Descent are generated dynamically in order to test the model's resistance against gradient based attacks. We opt to use slightly stronger parameters to create more adversarial pressure and attempt to maximize loss more aggressively while still regularizing the gradients to avoid total semantic loss. We used an alpha value of 1, epsilon clipping threshold of 0.12, and max iterations of 50. (Previous values used for training perturbations were 1, 0.1, 20 respectively). The perturbations were made on the testing set which is kept constant as control while the gradient attacks are generated with dependency on the specific model used as the defender.

We measure the accuracies at inference time for each of the models against each of the data sets and discuss our findings in the next section.

5 Results

We evaluate our models using binary prediction accuracy. We gauge the efficacy of our models by returning the percent of correct sentiment predictions.

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \times 100$$

5.1 Datasets

The main scope of our work is to classify sentiment in examples where natural language tendencies are not necessarily adhered to. For that reason, we opted to use a dataset that is more representative of slang and shorthand of the English language, as well as a corpus that is likelier to already contain typos. We used the Sentiment140 Kaggle Dataset (Kazanova, 2021), which contains Tweets from the social media site Twitter(now known as X). The dataset contains 1.6 millions Tweets posted from the late 2000s, and each Tweet has a binary label for its sentiment classification, either positive or negative.

Perturbed Datasets Our goal was to train and adapt BERT so that it is robust against adversar-

Comparison of Accuracy of Models on Various Datasets			
Testing Dataset	**Original Tweets**	**Typoed Tweets**	**PGD Tweets**
BERT	85.5	68.5	66.5
Typo RoBERT	83.5	70.5	70.0
PGD RoBERT	81.9	65.5	74.0

Table 1: Comparison of Accuracy of Models on Various Datasets

ial inputs. In order to do this, we had to concoct datasets that were laden with adversarial inputs. We were able to create these datasets by realistically manipulating the existing training data.

Typo Dataset As we gallivant through the digital age, the transmission of messages via the internet is exploding, with tens of billions being sent a day as of 2021 (Council, 2021). However, as messages can be sent easier and more readily, there is a higher opportunity cost to fixing typos, as now messages can be sent at a rate where they closely mimic real life conversations. It follows that fixing typos is inefficient, and moreover, they can be ignored as messages are much more casual.

Our formulation of this dataset with typos follows a somewhat naive approach. For each letter on the keyboard, we first obtain a set of swappable characters based on proximal keys.

Then, we iterate through the characters of the Tweet, randomly swapping letters for a nearby letter at a frequency of 15%, which we found to be a realistic rate.

We performed this operation for both the testing and training dataset. For training, we introduced typos into 10,000 Tweets of the original dataset. We then added these 10,000 perturbed Tweets to 50,000 unchanged Tweets to complete the typo dataset.

Predicted Gradient Descent Dataset In contrast with our dataset composed of typos, this dataset takes advantage of the model's weaknesses.

Using Projected Gradient Descent (PGD), this dataset exploits the training method used for BERT and finds which words, when substituted, will yield the greatest amount of loss, while still maintaining the original sentences meaning.

Similarly to our construction of the typo dataset, we perturb 10,000 of the Tweets using this method. We then also add 10,000 original exams to complete the PGD dataset with 20,000 total examples. We use a greater proportion of perturbed Tweets in this scenario since we want the model to learn the adversarial examples now, because these adversarial example are derived and are not based off of randomness like the typo dataset.

5.2 Baselines

For our baseline model, we used BERT (Devlin et al., 2019), a revolutionary deep learning architecture geared towards natural language processing. BERT stands for Bidirectional Encoder Representations from Transformers, and when introduced it vastly improved upon previous models. Within NLP, it excels at sentiment analysis, which is what we explore. For our task, we use a pretrained BERT model and train it on our various training datasets to tailor it to our task. This strategy is based off the notion that BERT learns the semantics of language, and can then be tweaked towards a specific downstream task.

The model we compare our models to is the pretrained BERT model tailored to the original Tweet Kaggle dataset. We train this model with a batch size of 16 using an AdamW optimizer, and a learning rate of 2e-5 (0.00002). This training style is typical to using BERT for language tasks.

5.3 Experimental Results

In this section, we discuss the results of our experiments. As mentioned previously, we present two versions of RoBERT, both adversarially trained versions of BERT. We train our baseline model by training it on the original Tweet dataset, and we train the robust versions on the perturbed datsets. We train each model for 5 epochs, with a batch size of 16 using an AdamW optimizer, and a learning rate of 2e-5 (0.00002).

We trained the typo RoBERT on the typo dataset, which is composed of 10,000 Tweets with typos, and 50,000 original, unperturbed Tweets. We trained the PGD RoBERT on the PGD dataset, which is composed of 10,000 Tweets attacked with GPD, and 10,000 original, unperturbed Tweets.

We then tested each model on three datasets, reflecting the adversarial attacks. The first dataset is the original dataset, and the other two are the dataset with typos and the dataset with PGD

attacks. We tested each model with 200 examples of each dataset, reporting its classification accuracy by comparing the predicted label with the true label.

The version of RoBERT trained on the dataset containing typos performed well in all three datasets. It had a slight dropoff from the original BERT-base model in the original dataset, but performs better in the test dataset with typos. For testing against PGD attacks, the typo model showed an improvement over the original model. This indicates that although the typo training dataset was constructed on random typos, it was able generalize well and defend against more targeted attacks.

After testing RoBERT against gradient-based attacks we see a significant improvement in performance compared to BERT-base when dealing with adversarial attacks (Projected Gradient Descent). Because the test set examples are unseen and the perturbations are dynamically generated based on the loss of the model defending, our findings point towards evidence that RoBERT is not only robust against unseen attacks but also much more resistant against gradient based attacks created specifically to target its weaknesses. However, it performed the worst out of the three models against the typo test set.

Although neither of our models perform as well as BERT-base in the original sentiment classification task, they still achieve an accuracy in the ballpark of the BERT-base, meaning they do not excessively overfit the adversarial data.

Table 1 presents the results of our experiments. As we observe from the table, all 3 models perform significantly worse on the perturbed datasets. However, we also see that each model performs best in the task it was trained on. This is not surprising, since each model was trained to fit the training data that was prepared in the same way as the respective test set.

6 Conclusions and Future Work

In this paper, we propose two robust variations of BERT, RoBERT. We produce these two models by training a baseline BERT model using adversarially generated datasets. In evaluating our models, we see that they outperform the baseline model in their respective tasks, while still maintaining good accuracy on non-adversarial data as well. Our experimentation shows that by training BERT on adversarial data, we can obtain a model that is robust against similar attacks.

In our future work, we can make BERT more robust by incorporating features that allow it to adapt to new slang, or words that are prevalently used in day-to-day talk but are not yet recognized by datasets. Furthermore, we can also train BERT against additional adversarial attacks in order to improve its robustness. These attacks might take simliar form to the ones described in this paper, but can also be geared towards specific domains.

We note that in our model, we utilize a QWERTY keyboard to derive typos for our perturbed dataset. This keyboard is commonly used for the English language, which is the main scope of our work. But other languages use other keyboards, so the letter substitutions that may occur in a typo would be different since keymappings are different. We can extend our typo-based model to other languages by deriving typos based on the keyboard that is intrinsic to a language.

It's likely that there is some underlying relationship between a word and typos. We generate typos randomly based on nearby keys to a letter. Instead of this, we can utilize this underlying distribution to substitute typos in for the original words. This would give our model a better chance at fitting real world data than our current way of generating typos.

Further down the line, we can add readability and interpretation to our model. This may take the form of recognizing which word was changed, or which typos affected the meaning of the Tweet and lead to a misclassification.

At the end of the day, we took BERT, a renowned natural language processing model, and utilize adversarial training methods to create RoBERT, a model used for robust sentence classification. We prepare training and testing sets to explore and validate our model's performance, and find that it achieves success in defending against adversarial attacks, specifically in randomly generated typos and directed word-substitution attacks. In the future, we can add more features to this model to improve and expand its use, from obtaining better adversarial data-generation methods to supporting more languages. With the relevance of NLP in the current era, we expect robustness to be a valuable

tool for constructing these models.

7 Limitations

There are a few limitations in our work. In the case of our generated dataset with typos, sometimes the typos introduced changed the true sentiment of the Tweet. For instance, consider the Tweet *"I'm not interested in deep learning."* This is a sentence with negative sentiment, and the dataset would have it labelled accordingly. With introducing typo perturbations, the Tweet may be morphed into *"I'm not interested in deep learning."* This new sentence now has a positive sentiment. However, if BERT is trained with this perturbed example, it will maintain the old label. This could be avoided by revisiting the typo dataset, and relabelling any Tweets that have had their true sentiment changed.

Another limitation of the typo dataset is that occasionally the perturbation can remove any meaning at all for the sentence. At this point, the datapoint essentially becomes noise since it no longer has any sentiment. In 3, we can see that the BERT model's performance decreases as the typo frequency increases. This is expected, but it also illustrates how typos can quickly remove any semblance of the original sentence. We can minimize the effect of

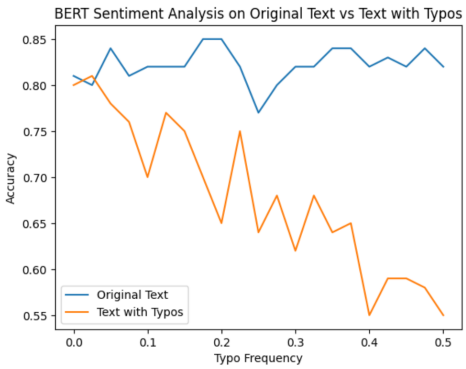

Figure 3: BERT's performance with varying rates of typos

this by inspecting the perturbed Tweets and removing any that have become gibberish.

As for the dataset, since it was obtained from Twitter (now known as X), there are a plethora of mentions (when people tag another user, in the form *"@username"*) present in Tweets. Additionally, there are also many links to other media in these Tweets. These two entities do not represent words in language, so they clutter the real words and can distract the model since they do not have real meaning. To improve upon this limitation, we could screen the Tweets and filter out such terms. However, doing this may remove certain context and affect positioning of the words, since they usually function as nouns within the Tweets.

The only dataset that was involved in our training was a dataset composed of Tweets from Twitter (now known as X). This approach gave our model an edge in language that is geared to conversational and one-sided, but limits its generalizability towards other forms of language. We can increase the strength of our model by utilizing other, more datasets that introduce different styles of language. For instance, we could also train it on emails or office memos. Although this would come at a greater computational cost, it would likely increase the efficacy and robustness of the model.

Most of the Tweets in the dataset were posted in 2009. The dataset is nearly 15 years old, so by using an older dataset it may fail to capture recent changes in the English language. Such changes may include new vocabulary, new sentence structure, or different meaning of words. For instance, the term "sick", at one point, had an entirely negative connotation. However, in the present that term can now have positive connotations such as in the phrase *"That skateboard is sick!"* We can improve this aspect of our model by acquiring a newer, more up-to-date dataset that includes more uses of new terminology and structure. However, this is difficult to obtain this data, especially since language is always evolving.

In our work, we only used the BERT model. There are other powerful models in NLP, and some may perform better in being robust in adversarial attacks. Although exploring another model would require more time, research, and computational resources, it would be beneficial to experiment with a new model in search for another architecture to improve on RoBERT with. Aside from obtaining a model to improve upon RoBERT, it would improve the validation of our results if we had another robust model to compare our performance to. We performed most of our work in Google Colaboratory, or Colab for short. Although it was effective for coalescing effort on this project, it did present limitations. For one, Colab was dependent on a network connection in order to run, meaning that it did not permit remote work. Additionally, Colab throttles hardware usage, which sometimes impeded methods that required longer training times and more use of computational power. If we used a service

with more computational accessibility, we would have been able to train our model for more epochs and achieve a more powerful and better performing model.

References

Nicholas Boucher, Ilia Shumailov, Ross Anderson, and Nicolas Papernot. 2021. Bad characters: Imperceptible nlp attacks.

Forbes Tech Council. 2021. The past, present, and future of messaging. Forbes.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples.

Chuan Guo, Alexandre Sablayrolles, Hervé Jégou, and Douwe Kiela. 2021. Gradient-based adversarial attacks against text transformers.

Kazanova. 2021. Sentiment140 kaggle dataset. Kaggle.

Haoran Li, Yulin Chen, Jinglong Luo, Yan Kang, Xiaojin Zhang, Qi Hu, Chunkit Chan, and Yangqiu Song. 2023. Privacy in large language models: Attacks, defenses and future directions.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. Bert-attack: Adversarial attack against bert using bert.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2019. Towards deep learning models resistant to adversarial attacks.

Tarek Naous, Michael J. Ryan, Alan Ritter, and Wei Xu. 2023. Having beer after prayer? measuring cultural bias in large language models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2020. Xlnet: Generalized autoregressive pretraining for language understanding.

Gokul Yenduri, Ramalingam M, Chemmalar Selvi G, Supriya Y, Gautam Srivastava, Praveen Kumar Reddy Maddikunta, Deepti Raj G, Rutvij H Jhaveri, Prabadevi B, Weizheng Wang, Athanasios V. Vasilakos, and Thippa Reddy Gadekallu. 2023. Generative pre-trained transformer: A comprehensive review on enabling technologies, potential applications, emerging challenges, and future directions.