Is Prefix tuning robust?

Parth Shettiwar
Computer Science Department,
University of California, Los Angeles
parthshettiwar@g.ucla.edu

Abstract

The recent years have seen a rise of large number of models where the attempt is to save the storage space owing to large number of trainable parameters of Deep Neural Networks (for example 175 billion parameters in (Brown et al., 2020)). One such light-weight alternative work is prefix tuning (Li and Liang, 2021), where, as compared to traditional fine-tuning, we train only a task-specific prefix and rest backbone model parameters remain same. As a result, this results in huge decrement in storage requirements and hence less complexities while deploying the models on space constraint devices for real time applications. Furthermore, as natural language models become more widely used in real-world applications, the risk of adversarial attacks by malicious actors has also started increasing. In past there have been multiple works to evaluate the robustness of standard NLP models. However, there has been no study in past to understand the same on the lightweight alternatives, particularly the recent work on prefix-tuning (Li and Liang, 2021). As part of this work, we try to understand and explore the robustness of BERT-based prefix-tuned models where we particularly expose the prefix tuned models to adversarial attacks, namely (Jin et al., 2020) and (Li et al., 2020), and see their behaviour as compared to baseline fine-tuned models. The experiments indicate that there is no definite conclusive evidence and the robustness comparison depends heavily on which dataset the models are trained on. We have made our codebase publicly available at `https://github.com/parth-shettiwar/Robustness-of-Prefix-Tuning`.[*].

1 Introduction

Machine Learning models have gained amazing success in numerous Natural language processing tasks such as sentiment analysis, language translation, and so on in recent years. However, they have lately been proven to be vulnerable to adversarial instances, which are legitimate inputs that have been altered by subtle and often unnoticeable perturbations. These carefully prepared examples can trick a target model, where it would output incorrect predictions or translations, raising severe questions about the integrity of present ML systems (Kurakin et al., 2017). Overall the goal of adversarial attack is to minimize the perturbation on input side but at same time maximize the probability of model outputting incorrect prediction. For example, in a classification task, this would boil down to the model outputting incorrect label for a minor change in text on the input side.

Recently, there have been studies on evaluating the adversarial robustness of fine-tuned models. It is important to note that fine tuning of a model results in catastrophic forgetting, since a complete retraining of model is done. This might lead to a poor adversarial robustness owing to forgetting the underlying original dataset distribution used for training the model. At same time, fine-tuning process also might overfit to the training data and reduces the model's ability to generalize to new inputs. For example, in a study by (Niu and Bansal, 2018), fine-tuning was shown to decrease the robustness of a pretrained model to adversarial attacks on NLP tasks.

One major drawback of fine-tuning of a model is that a full copy of the model needs to be saved everytime the model is trained for a particular task. Lightweight alternatives to this like (Li and Liang, 2021) have been recently introduced. Prefix tuning involves adding a prefix to the input text during fine-tuning to condition the model on a specific task. The prefix can be a sequence of tokens that specify the task, such as a classification label or a prompt, and is added to every input text instance during fine-tuning. This allows the model to learn task-specific features while retaining its ability to generate coherent and grammatical text. While tuning a model

[*] The work has been done in a collaborative fashion by 3 people, where each of them have worked on one dataset

for a downstream task, the model's parameters are frozen, and the task-specific vector's parameters are optimized solely. Prefix tuning has been shown to be effective in improving the performance of language models on a wide range of tasks, including text classification, question answering, and natural language generation. The catastrophic forgetting is also resolved since we directly leverage the fine tuned model weights.

Considering this resistance to catastrophic forgetting, the prefix tuned model might actually retain the underlying data distribution and hence might as well be more robust to various adversarial attacks. However, there has been no study on this in the past. In this work, we aim to perform a fair comparison of the robustness of fine tuned models with the prefix tuned models. More formally, our contributions through this work are:

- Comprehensive study to evaluate the robustness of prefix tuned models and compare it with the respective baseline fine tuned models on imdb dataset [†]

- Training fine tuned and prefix tuned models on downstream task of classification and later attacking them using BERT and TextFooler attack, (Li et al., 2020; Jin et al., 2020) to evaluate their adversarial accuracies

- Analysing the clean and adversarial accuracies while varying the budget of number of queries that a particular attack can use.

The experimental results suggest that finetuned models do tend to have more robustness to adversarial attacks than prefix tuned models but there is no conclusive evidence for this. As we see, this is highly dependent on which dataset we are using for the experiments. Furthermore, for a particular dataset also, there is significant difference in robustness between a finetuned and prefix tuned model.

2 Related Work

We divide this section into three parts: 1) Adversarial Attacks in NLP, 2) Natural Language Model Fine-tuning, 3) Lightweight alternatives in NLP

2.1 Adversarial Attacks in NLP

The adversarial attacks in NLP can be divided based on whether they are character level or word level or sentence level. Furthermore, they can be either black-box or white box depending on how much knowledge the attacker has pertaining to the model in hand. The overall strategy is to perturb the input, which is usually human imperceptible (minimal perturbation) and ensure that the output given by the model changes. This can be done by applying gradient descent over continuous space of images but is relatively difficult for discrete space like text (recently explored by (Papernot et al., 2016), (Ebrahimi et al., 2018)).

Character level synonym substitution has been done in past by (Ren et al., 2019). Manual replacement of words is done by (Glockner et al., 2018) to preserve the semantic information. A more coarser technique where modifications are done at the level of phrase was explored by (Liang et al., 2017). However this method produces sentences which are relatively less fluent and has inconsistencies. Word level perturbations based on glove based embeddings (Pennington et al., 2014) was done by (Li et al., 2019)

As part of this work, we have mainly focused ourselves on 2 adversarial attacks, (Li et al., 2020) and (Jin et al., 2020). In BERT attack (Li et al., 2020), pre-trained language model is used, as the name suggests, BERT (Devlin et al., 2019a), to generate adversarial examples. In this way, it can use the knowledge learned by the pre-trained language model to generate perturbations. In addition, as compared to other attacking algorithms, BERT-Attack has a greater attacking success rate and a smaller perturb percentage with less access numbers to the target model. On other hand, (Jin et al., 2020) is a substitution based adversarial attack where words in the sentence are replaced by their synonyms but the syntactic meaning is preserved during this. TextFooler uses a combination of gradient-based optimization and heuristic rules to generate perturbations that maximize the difference between the predicted output of the target model for the original and perturbed text. Preservation of syntactic meaning and overall simplicity of this attack, were the main reasons for choosing this attack for all the experiments.

2.2 Natural Language Model Fine-tuning

Fine-tuning a language model entails training it with new data from your domain. During this procedure, the weights of the previous model are adjusted to account for the peculiarities of the new

[†] Taken from https://huggingface.co/datasets

data and the task at hand. Current cutting-edge methods for natural language generation rely on fine-tuning of pretrained language models. For example, (Kale and Rastogi, 2020) fine-tunes a sequence-to-sequence model for table-to-text generation, whereas researchers use masked language models (Devlin et al., 2019b) and encode-decoder models (Lewis et al., 2020) for summarization tasks. Fine-tuning is also employed for other tasks including machine translation and dialogue synthesis (Zhang et al., 2020) (Cooper Stickland et al., 2021). Works like (Liu et al., 2020) were also done where the hyper-parameters were perfectly tuned to improve the overall accuracy over the standard Bert model. On a different tangent, works like (Raffel et al., 2020) focused on achieving better accuracies using transfer learning approaches, where most weights are kept fixed during training of the model.

In this work, we have specifically used a standard BERT model to demonstrate its effectiveness on typical classification tasks. However, the same approach can be implemented with other models and for various other generation tasks.

2.3 Lightweight alternatives

Although fine tuning is effective, it often comes with significant space demands. There are mainly 3 types of approaches considered to address this problem, where each approach avoids saving the complete copy of weights for each task, and saves only a few important parameters which are task specific while fixing many other weights between tasks.

The first is lightweight fine-tuning (Zhang et al., 2019), (Houlsby et al., 2019), (Radiya-Dixit and Wang, 2020), where small segments of the network architecture are allowed to be changed during training and rest all weights are frozen. The second is prompt-tuning (Brown et al., 2020), (Sun and Lai, 2020) where the input is prepended with some context, to steer the output to a desired value. The complete pre-trained model is kept fixed and only small number of task specific tokens are saved. This brings down the total storage requirements by a huge margin. Finally, (Li and Liang, 2021) introduced the idea of prefix tuning which takes one step over prompt tuning where instead of saving real tokens, continuous task-specific vectors are introduced as prefix to the input context. This improves the performance by a huge margin.

In this work, we have investigated the robustness of the prefix-tuning approach (Li and Liang, 2021). Similar experiments can be performed to evaluate the robustness of other light weight tuning approaches.

3 Problem Statement and Methodology

Given a trained prefix tuned model, we aim to establish its degree of robustness by performing various adversarial attacks. To serve as a standard basis for comparison, we also attack the models trained using a standard fine-tuning approach.
From our experiments, we first train the BERT models on the complete training dataset using both fine-tuning and prefix tuning approaches on a downstream task of classification, with an optimal learning rate to maximize the clean accuracy (We choose classification task as it is easy to use and infer the results from). This is done for 3 seeds and hence we get 6 models (3 each for fine tuning and prefix tuning). We then select a random sample of 500 text inputs and their associated labels for our adversarial attack experiments. Leveraging the BERT-Attack and Textfooler attack algorithms, we then attack each model for 4 different query budgets and note down the resulting adversarial accuracies.

4 Implementation Details

4.1 Fine-Tuning and Prefix tuning

The huggingface[‡] and adapter-hub[§] libraries were used for fine-tuning and prefix-tuning training of BERT, respectively. The huggingface-provided `bert-base-uncased` is used as the base model. The batch size is set to 128 and the models are trained for 50 epochs. For each dataset and model, learning rates are tuned appropriately, so that both prefix tuned and fine tuned models achieve their optimal accuracies. NVIDIA A-100 and RTX 2080 GPUs are used to train the models.

4.2 Bert and Textfooler Attacks

To carry out the adversarial attacks, we leveraged the textattack package (Morris et al., 2020) which is a Python framework supporting mainly 3 types of operations: adversarial attacks, data augmentation and model training. This library makes experimenting with the robustness of NLP models simple and fast, considering the various knobs we can use to

[‡] https://huggingface.co/
[§] https://adapterhub.ml/

tune our attack and direct integration in the existing setup.

For the BERT-attack, a vanilla Textattack BERT-Attack was performed, with the exception of the query budget being varied between experiments and the maximum candidates parameter being set to 3. This was done to limit the run-time for passages that contain many sub-words. For the textfooler attack, we only changed the query budget while running the attacks. For our experiments, we set the query budget in such a way that the adversarial accuracy was between 80% and 30%. Due to limited computation power, a subset of 500 random examples were considered from the dataset for performing the adversarial attacks. These examples remained consistent across different models and seeds.

4.3 Dataset Details

We consider the imdb dataset to perform all our experiments. imdb is a large Movie Review Dataset. It is a collection of data used for determining whether a movie review has a positive or negative sentiment. The dataset includes 25,000 movie reviews with strong polar opinions for training and another 25,000 for testing.

5 Experimental Results

5.1 Results

In this section, we present the clean accuracies of the models trained using both prefix tuning and fine tuning. We also show plots comparing the adversarial accuracy against the average number of queries. Seed A is the default seed (unknown), Seed B is seed 23 and Seed C is seed 24.

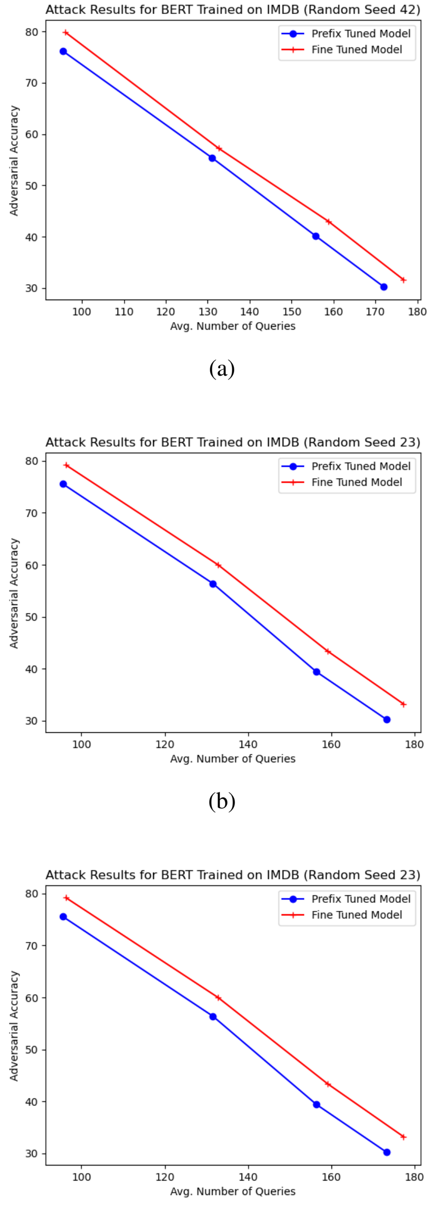
Method	Seed A	Seed B	Seed C
`finetuning`	91.4	90.6	90.2
`prefix-tuning`	90.6	90.0	89.2

Table 1: Table showing the clean accuracies of the trained models on imdb using different approaches

The following figures show the comparison between the finetuned model and the prefix-tuned model in terms of adversarial accuracy.

5.2 Implications

[¶] Firstly, from the table we observe that the clean accuracy of prefix tuning is just less than clean



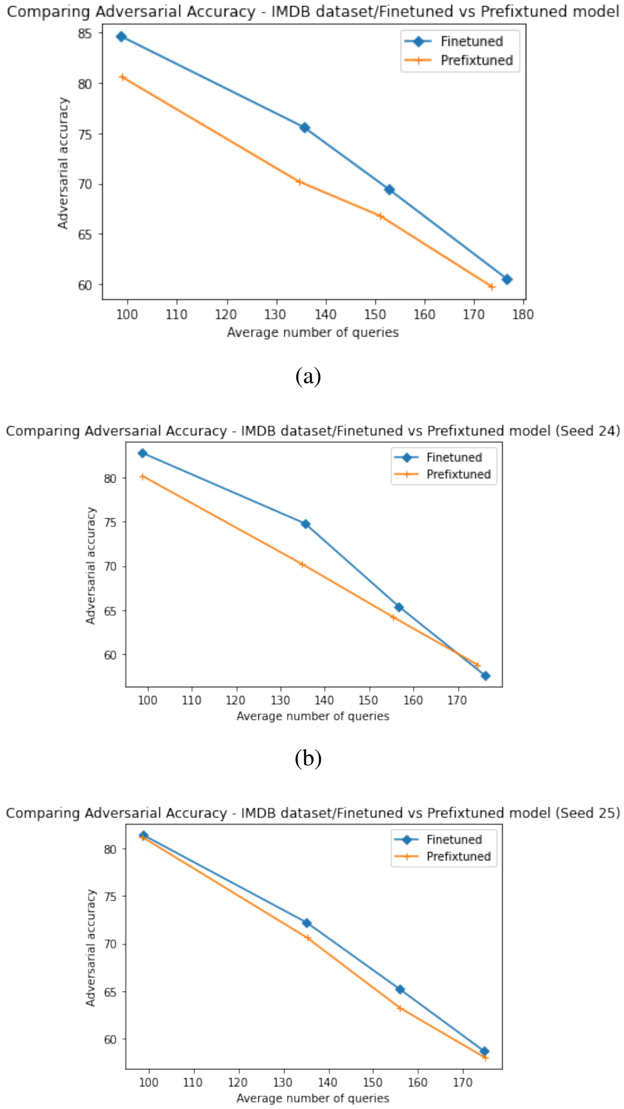
(a)

(b)

(c)

Figure 1: Figure showing the adversarial accuracy vs average number of queries for BERT-attack for the imdb dataset

[¶] This work was done in a group of 3, where other group members worked on `sst2` and `ag_news` datasets. The implications are written after including those results too



(a)

(b)

(c)

Figure 2: Plots showing the adversarial accuracy vs average number of queries for Textfooler attack for the imdb dataset

accuracy of fine tuning approach. This trend is observed across all the datasets. This makes sense, since we are using a fraction of parameters in prefix tuning as compared to the fine tuning approach. Next from the above plots it is clear that fine-tuning outperforms the prefix-tuning on both the attacks and so is the case when we conducted this experiment for sst2 dataset. However, as we see, this is not the case always, since in ag_news dataset, the observations are opposite. As a result, it would not be correct to make any general statements about one method being better than the other (in terms of robustness) as the results are mixed. The question of prefix tuning or fine tuning being more robust depends highly on which dataset we are working on. Also once the dataset is fixed, we observe that there is lot of variation in the adversarial accuracy as the budget of average number of queries is varied (for example, in both imdb and ag_news datasets, at some point of variation in budget, we see the reversal in performance of prefix tuning and fine tuning). At the same time, it is also observed that, in general, the gap between the performance of these two methods is maintained with variation of the budget for a particular dataset. Overall, there is no conclusive evidence on which method outperforms other in terms of robustness with respect to (Jin et al., 2020) and (Li et al., 2020) attacks.

6 Conclusion

In this work, we investigated the adversarial robustness of the standard fine-tuning and prefix tuning approaches with base as the BERT model. To achieve this, we run experiments on several sequence classification datasets and then expose the model to adversarial attack techniques such as BERT-Attack and Textfooler. Through the experiments, it is observed that, while no definite rule can be discovered, there are numerous relevant trends to be observed. Fine-tuned BERT is often more robust than prefix tuned BERT. Also, within a particular dataset, robustness tends to be better for one type of model than the other.

This work serves as a foundation and a call for further research in this domain. As part of future work, we can conduct robustness experiments while we control the various other knobs like prefix length. The prefix length was fixed to 30 in all the experiments but is an important hyperparameter in the training of prefix-tuning of the models and varying that parameter might also uncover some more trends. Running the adversarial attacks in some different constraint settings like fixing the percentage of words perturbations, could also be done. Furthermore, the whole experimental setup was done in a very constraint setting due to limited resources and time, where we had fixed the base model and restricted ourselves to 2 adversarial attacks. Variation along these ends would definitely reveal new trends in this research. Finally, throughout this work, we focused ourselves on only one of the lightweight alternatives which is prefix tuning, but as we see in the literature review, there are multiple other methods in this domain like (Zhang et al., 2019), (Houlsby et al., 2019), (Brown et al., 2020) and would be of interest to further check out their robustness as compared to standard fine-tuning. Overall this work is a starting point for further research in this domain, which would directly affect the securities and integrity of practical ML systems and will be of utmost importance to understand the efficacy of deploying the various lightweight alternative models in real world applications.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Asa Cooper Stickland, Xian Li, and Marjan Ghazvininejad. 2021. Recipes for adapting pre-trained monolingual and multilingual models to machine translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3440–3453, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019a. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

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

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

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking NLI systems with sentences that require simple lexical inferences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 650–655, Melbourne, Australia. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT really robust? A strong baseline for natural language attack on text classification and entailment. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8018–8025. AAAI Press.

Mihir Kale and Abhinav Rastogi. 2020. Text-to-text pre-training for data-to-text tasks. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 97–102, Dublin, Ireland. Association for Computational Linguistics.

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

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2019. Textbugger: Generating adversarial text against real-world applications. In *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019*. The Internet Society.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. BERT-ATTACK: adversarial attack against BERT using BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6193–6202. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4582–4597. Association for Computational Linguistics.

Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. 2017. Deep text classification can be fooled. *ArXiv*, abs/1704.08006.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Ro{bert}a: A robustly optimized {bert} pretraining approach.

John X. Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, pages 119–126. Association for Computational Linguistics.

Tong Niu and Mohit Bansal. 2018. Adversarial oversensitivity and over-stability strategies for dialogue models. In *Conference on Computational Natural Language Learning*.

Nicolas Papernot, Patrick D. McDaniel, Ananthram Swami, and Richard E. Harang. 2016. Crafting adversarial input sequences for recurrent neural networks. In *2016 IEEE Military Communications Conference, MILCOM 2016, Baltimore, MD, USA, November 1-3, 2016*, pages 49–54. IEEE.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL.

Evani Radiya-Dixit and Xin Wang. 2020. How fine can fine-tuning be? learning efficient language models. In *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*, pages 2435–2443. PMLR.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *Annual Meeting of the Association for Computational Linguistics*.

Fan-Keng Sun and Cheng-I Lai. 2020. Conditioned natural language generation using only unconditioned language model: An exploration. *CoRR*, abs/2011.07347.

Jeffrey O. Zhang, Alexander Sax, Amir Roshan Zamir, Leonidas J. Guibas, and Jitendra Malik. 2019. Side-tuning: A baseline for network adaptation via additive side networks. In *European Conference on Computer Vision*.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

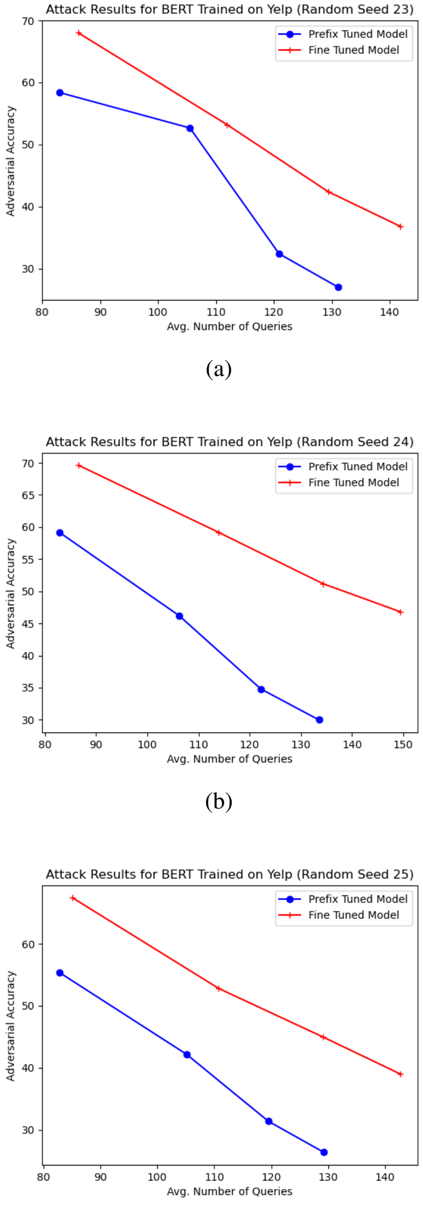
7 Appendix

We also conduct experiments on another dataset `yelp_polarity`. Following are the clean accuracies for different seeds:

Method	Seed A	Seed B	Seed C
finetuning	95.0	95.4	95.2
prefix-tuning	93.8	94.6	94.0

Table 2: Table showing the clean accuracies of the trained models using finetuning and prefix tuning

Following are the plots attached with respect to the 2 attacks. As we see the observations remain similar, where fine tuning tend to outperform the prefix tuning on both attacks.

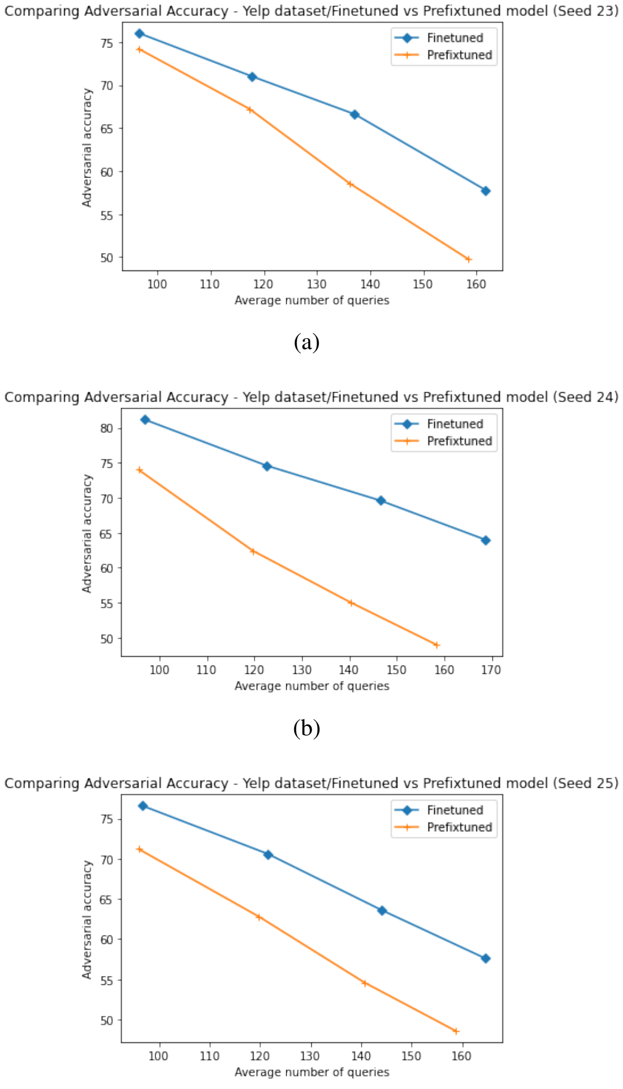


(a)

(b)

(c)

Figure 3: Figure showing the adversarial accuracy vs average number of queries for BERT-attack for the yelp dataset



(a)

(b)

(c)

Figure 4: Plots showing the adversarial accuracy vs average number of queries for Textfooler attack for the yelp dataset

7.1 Model Tuning Hyperparameters

Note: All hyperparameters were tuned on imdb dataset.

7.1.1 Prefix Tuned Model

- Prefix length: 30
- Learning rate: 1e-5
- Batch size: 128
- Number of training epochs: 50

7.1.2 Fine Tuned Model

- Learning rate: 5e-6
- Batch size: 128
- Number of training epochs: 50

Adversarial Attack Hyperparameters

7.1.3 BERT-Attack

- Maximum candidates: 3
- Sentence similarity threshold: 0.2

7.1.4 Textfooler

- Sentence similarity threshold: 0.84