Multi Task Fine Tuning of BERT Using Adversarial Regularization and Priority Sampling

Stanford CS224N Default Project

Chloe Trujillo
Department of Computer Science
Stanford University
chloe818@stanford.edu

Mohsen Mahvash
Department of Computer Science
Stanford University
mahvash@stanford.edu

Mentor: Annabelle Tingke Wang
No outside mentor

Abstract

This study utilizes the Adversarial Regularization technique alongside adaptive priority sampling of task databases to enhance the efficacy of the BERT BASE model across three distinct tasks: sentiment analysis, paraphrase detection, and text similarity regression. The Adversarial Regularization method augments the standard loss function with an additional term, mitigating the classifier model's response to minor input embedding variations fed into the BERT model. This regularization mechanism combats overfitting. Additionally, adaptive priority sampling dynamically adjusts the data stream utilized for training optimization, considering factors such as real-time task performance metrics for training and development sets, and the proportional size of each dataset.

1 Introduction

In the realm of machine learning, particularly within natural language processing (NLP), traditional deep learning methods heavily rely on extensive labeled datasets for supervised learning. However, acquiring such large amounts of labeled data can often be impractical or unfeasible in real-world scenarios. A promising alternative approach has emerged, centered around leveraging pre-trained models that reduce the dependency on labeled data for initial training. These pre-trained models can be fine-tuned or adapted using smaller sets of labeled data tailored to specific tasks.

Recent advancements have led to the development of several pre-trained language models, such as BERT Devlin et al. (2018) and GPT-3 Radford et al. (2019), which have been trained on extensive datasets. These pre-trained models serve as powerful tools in NLP, allowing for more efficient utilization of labeled data. However, fine-tuning on limited data presents the risk of overfitting, where the model overly tailors itself to the training data, potentially compromising its performance on new, unseen data. To address this, various regularization techniques can be applied.

Another critical aspect in NLP is multitasking, which holds great promise in enhancing efficiency, robustness, and adaptability in addressing diverse linguistic tasks. Multitasking models involve training a single model to perform multiple NLP tasks simultaneously, leveraging shared knowledge and representations across tasks. This approach not only promotes improved generalization but also optimizes resource utilization by eliminating the need for separate models for each task, making it particularly valuable in resource-constrained environments.

In this paper, our objective is to explore regularization and sampling techniques to develop a multitask BERT model tailored for three distinct tasks: sentiment analysis, paraphrase detection, and text similarity regression. Our primary focus is on enhancing performance through the implementation of adversarial regularization techniques and adaptive round-robin sampling of databases of the tasks. By

employing adversarial regularization, we aim to smooth the model's responses, thereby improving its generalization across tasks. Additionally, the use of adaptive sampling during training aims to enhance performance uniformly across all tasks.

The efficacy of our approach will be evaluated using average metrics for each task. Specifically, we will use accuracy for sentiment analysis and paraphrasing tasks, while employing Pearson correlation for the similarity task.

2 Literature Review

Regularization techniques, which discourage the model from becoming overly complex and specific to the training data, are crucial for preventing overfitting. Dropout, introduced in 2014 to prevent neuron co-dependence, has since become a standard practice in many language models (SOURCE). Adversarial regularization, as introduced by Miyato et al. Miyato et al. (2018) in 2018, adds noise to training data to smooth the model's responses and improve its generalization across computer vision tasks. This method was later adapted by Jiang et al. Jiang et al. (2020) for specific language tasks, which we leverage in our project to generalize across multiple tasks.

Multitask training shows promise in enhancing efficiency, robustness, and adaptability in addressing various linguistic tasks. Similar to our baseline BERT implementation, Bi et al. Bi et al. (2022) propose multi-task learning, where losses from different tasks are combined. We employ this approach for multitask fine-tuning, along with round-robin sampling, where each task's training data takes turns to prevent one task from dominating the training process. Further, Sun et al. Sun et al. (2019) explored advanced pretraining and fine-tuning methods to adapt BERT for text classification; they found that incorporating lexical, syntactic, and semantic information through continual pretraining, along with specific learning rate adjustments, led to performance enhancements that we have demonstrated extend to multi-task models.

Yu et al. Yu et al. (2020) highlight a challenge associated with multi-task learning: conflicting gradient directions from different tasks. To address this issue, they introduce Gradient Surgery, a technique involving adjusting the gradients of one task by projecting them onto the normal plane of another task's gradient. Adaptive priority sampling of databases also presents a viable avenue for mitigating gradient direction discrepancies.

3 Approach

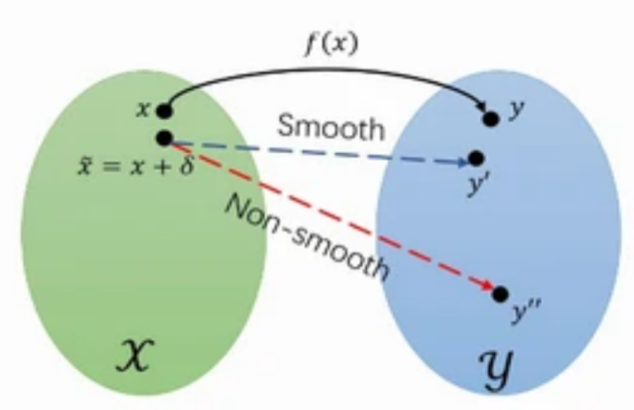

Figure 1: Illustration of the classifier model f mapping sentence embeddings x to probability vectors y. f is smooth if $x + \delta$ maps to y' and nonsmooth if $x + \delta$ maps to y''.

We utilize a BERT model as the foundation and construct three distinct classifiers on top of it. Each classifier uses the hidden states of the same BERT and produces predicted output for one of these three tasks: sentence sentiment classification (SST), paraphrase detection, and semantic textual similarity

(STS). Multi-task training is employed to fine-tune the BERT model, where losses from all tasks are aggregated.

For paraphrase detection and semantic textual similarity (STS), we explore two different model types:

1. Utilizing the MiniBERT architecture with default classifiers that feed two separate sentences to the BERT and compare output to predict the results.
2. Utilizing BERT from pytorch-transformers with a linear classifier where input sentences are concatenated as a single long sentence (denoted as `long_sentence = Sentence1 + "SEP" + Sentence2`). Mulyar (2022)

The primary focus of our approach revolves around two techniques: adversarial regularization to overcome overfitting to data, and priority sampling to minimize overfitting to a specific task.

3.1 Adversarial Regularization

Even with training the BERT model for a single-task classifier using dropout, we observe overfitting, especially when provided limited labeled training data. To mitigate this, we implement a smoothness-inducing regularization technique, introduced by Jiang et al. (2020), for each task separately.

Figure 1 depicts a smooth classifier model $f(x, \theta)$, where x represents the input sentence embeddings, and θ denotes the model parameters. This classifier is considered smooth concerning x if small noise/perturbations δ result in negligible changes to the output y.

The adversarial regularizer fine-tunes the model parameters θ to minimize a two-term loss function:

$$\text{loss}(\theta) = L(\theta) + \lambda_s R_s(\theta) \tag{1}$$

Here, $L(\theta)$ denotes the standard cross-entropy loss function, guiding the classifier to adhere to labeled data during training, while $R_s(\theta)$ represents the smoothness regularization term, ensuring the smoothness of the classifier f. $\lambda_s > 0$ serves as a tuning parameter to weight this regularization.

The smoothness regularization term $R_s(\theta)$ is expressed as:

$$R_s(\theta) = \frac{1}{n} \sum_{i=1}^{n} \text{distance}\left(f(x_i + \delta_i, \theta), f(x_i, \theta)\right) \tag{2}$$

where i indexes the training samples. The function distance$(\cdot)$ calculates the difference between two probability vectors (the output of normal input vs perturbed input) such as $||f(x_i + \delta_i, \theta) - f(x_i, \theta)||_2^2$.

We implemented the classifier ourselves from the Default Project Handout, and we incorporated the smoothing regularization by copying the SMART algorithm outlined in JIANG SOURCE with modifications to handle the dropout implemented in the baseline BERT. We also leveraged a smartpy-torch package from Archinetai (2024) which takes an input embedding and generates an adversarial embedding with noise which we set at 5%, to get a perturbed state that we implemented in a separate, perturbation-specific forward pass function in the BERT class. We then pass the concatenation of two input embedding to the package function for STS and paraphrasing tasks. Although we did not have the time or compute units to experiment with several different values of λ_s, we tried a few of the paper's suggested weights $\lambda_s \in 0.1, 1, 5, 10, 100$ to make sure our modification was being applied at the scale we imagined.
One thing we noticed is that the baseline dropout was injecting the same/more noise than the adversarial regularization noise, and was diminishing the learning capabilities as the dropout noise did not have a learnable loss. Thus, we tried disabling it, which yielded slightly better results.

3.2 Priority Sampling

Imagine a situation where one of the task databases contains significantly more samples than the combined samples of the other tasks, and this task with heavy data is trained last. In such a scenario, the final training step may substantially degrade the performance of other classifiers. This phenomenon can be illustrated by observing that the gradient of the final task in the training chain

can diverge significantly from the gradient of the earliest task in the training chain. Consequently, many steps of the last task may deviate significantly from the optimal solution of the earlier tasks. To mitigate these conditions, we have developed an adaptive round-robin strategy for picking the batches of samples sent for training.

During each epoch of optimization, we iterate over each task and select a p_i number of batches for each task i. The priority number p_i represents the rounded ratio of the training samples for task i normalized to the number of training samples for tasks with lower sample counts. p_i for task i varies within a range

$$N_i \geq p_i \geq 0$$

Here, N_i is the rounded ratio of the sample size of task i database to the size of the database with the smallest size.

At the beginning of training, we set $p_i = N_i$ for several epochs, and then during training, when the performance of task i saturates, we gradually decrease p_i even to zero. Once the performance metric of all tasks are saturated we activate adversarial regularization term and set all $p_i = 1$ to mitigate overfitting .

4 Experiments

4.1 Data

Table 1 describes the datasets used in our experiments. We only used the data provided to us by CS224n – These datasets encompass five tasks, including four classification tasks and one regression task.

Table 1: Experiment 2 Dataset Overview

Dataset	Description	Task	Type/Classes	Samples
SST-5	Sentiment Analysis	Movie review sentiment classification	Classification/5	12,855
QQP	Paraphrase Detection	Determining if question pairs are paraphrases	Classification/2	202,152
STS-B	Text Similarity	Measuring the similarity between sentence pairs	Regression/NA	8,628
SST	Sentiment Analysis	Movie review sentiment classification	Classification/5	11,855
CFIMDB	Sentiment Analysis	Sentiment classification of IMDB movie reviews	Classification/2	2,434

4.2 Evaluation method

We use accuracy to evaluate classification tasks of Table 1. Accuracy would indicate the percentage of correctly classified out of the total number of instances in the dataset. We use Pearson correlation to evaluate the regression task which is Text Similarity. This metric quantifies the linear relationship between the true values and the predicted values.

4.3 Experimental Details

All experiments were conducted with a learning rate of 1×10^{-5} and a hidden-layer dropout probability of 0.3. In our SMART implementation Jiang et al. (2020), specific hyperparameter values were set as follows: $\lambda = 5$, $\epsilon = 1 \times 10^{-5}$, $\sigma = 1 \times 10^{-5}$, $\beta = 0.995$, $\mu = 1$, and $\eta = 1 \times 10^{-3}$.

During training, each iteration processed $\sum$ batch size $\times\, p_i$ samples of data sets, where p_i represents the priority of each task. We used a batch size of 8 for SMART training and 16 for baseline training. Each epoch consisted of 1000 batches from all datasets. The value of p_i varied from 0 to 10 for paraphrase classification and set to 1 for other tasks.

We provide the details of five configurations2. For training our baseline, we assigns equal priority ($\Pi = 1$) to each task during training, resulting in an equal number of samples collected for each task. We also use type 1 model. In contrast, the second model utilizes priority sampling, with one batch of SST, one batch of STC, and eight batches of paraphrase results collected per iteration.

The third model incorporates adversarial regularization ($\lambda = 5$) on top of the training procedure of the second model, with an equal number of samples collected for each task during training.

The forth and fifth model use type 2 model.

4.4 Results

Table 2 presents the evaluation results for three models developed in Experiment 2.

Table 2: Model Evaluation for Experiment 2

Method	Overall	SST Accuracy	Paraphrase Accuracy	STS Correlation
BaseLine ($\P_i = 1$)	0.510	0.510	0.446	0.395
Type 1, Priority Sampling of (1,8,1)	0.655	0.512	0.746	0.414
Type 1, Priority Sampling + Adversarial	0.672	0.528	0.752	0.473
Type 2, Priority Sampling of (1,8,1)	0.742	0.524	0.789	0.742
Type 2, Priority Sampling + Adversarial	0.753	0.524	0.819	0.841

As of writing this report, our best-performing model achieved a rank of 55 on the dev set leaderboard.

Our results demonstrate a 2 to 10 percent improvement in accuracy when adversarial regularization is applied. This finding is consistent with the improvement reported in Jiang et al. (2020). However, it contrasts with our initial expectations, as we did not anticipate such a minor improvement in accuracy from adversarial regularization.

Our results show significant improvement for STS and Paraphrase task using model2.

5 Analysis

Our results indicate a discrepancy in Pearson correlation for Model Type 1 and Model Type2. To gain insight into this issue, we compared the accuracy and correlation separately calculated for the training dataset and development dataset 3.

The results shows accuracy and correlation for the training dataset are close to 1 for STS Correlation Sentiment Accuracy while the same metrics for development set are around 0.5. This discrepancy suggests significant over fitting. To further investigate the overfitting problem

Table 3: Model Evaluation Results

Task	Sentiment Accuracy	Paraphrase Accuracy	STS Correlation
Development Dataset	0.504	0.746	0.485
Training Dataset	1.000	0.770	0.931
Test Dataset	0.528	0.752	0.827

To address this overfitting concern, we implemented several strategies. Initially, we narrowed down the tasks to focus solely on the STS task and experimented with various regression parameters, including $\epsilon = 1e - 6$ and $\lambda = 0.01, 5, 10, 50$. However, none of these mitigated the overfitting issue.

Furthermore, we attempted to improve the STS accuracy by transitioning from the FCos (Feature-wise Cosine Similarity) method to Cross Attention. Regrettably, this approach failed to enhance STS accuracy, and the overfitting problem persisted. Our findings suggest that although SMART regularization marginally improved metrics by 2 to 5 percent, it was insufficient to alleviate overfitting. However, We found model type 2 results are significantly bettr than model1.

Additionally, we scrutinized the results reported by Jiang et al. Jiang et al. (2020) for the STS task. Interestingly, their experiments with SMART regularization demonstrated less than a 1 percent improvement in correlation for the STS tasks they evaluated.

6 Conclusion

In this study, we delved into the efficacy of employing adversarial regularization and adaptive sampling techniques to bolster the performance of a multitask BERT model across a spectrum of natural language processing (NLP) tasks. Through the integration of these techniques, our aim was to mitigate overfitting and enhance the model's generalization capacities across diverse tasks.

Our experimental findings yielded promising outcomes. Adversarial regularization resulted in a notable enhancement of 2 to 5 percent in accuracy across tasks, aligning with prior research as noted in Jiang et al. (2020). However, we encountered challenges related to an increase in the correlation value within the development dataset for the Semantic Textual Similarity (STS) task and Models of Type1.

To address the persistent overfitting issue On Model 1 types, we employed various strategies, including modifying the STS classifier and conducting parameter experiments. Despite these efforts, overfitting persisted, underscoring the necessity for further exploration into more robust regularization techniques.

Looking ahead, we advocate for deeper investigation into the overfitting phenomenon. Furthermore, exploring the impact of multitask learning architectures on model performance and generalization across a wide array of linguistic tasks could yield valuable insights for future research pursuits.

Contributions: Chloe and Mohsen made equal contributions to various aspects of this project, including coding, analyzing results, paying for Colab credits (lol), and conducting experiments.

7 Appendix

We conducted another set of experiment evaluating BERT and regularized BERT models for sentiment analysis. For this experiment we use SST dataset and CFIMDB dataset. The training is done only using STS task dataset.

Table 4: Model Evaluaion for Experiment 2

Task	δ **= 0%,** $\lambda = 0$	δ **= 5%,** $\lambda = 1$
SST Accuracy	0.271	0.311
Paraphrase Accuracy	0.403	0.405
STS Correlation	0.020	0.029
Overall	0.395	0.409

References

Archinetai. 2024. Smart-PyTorch: A Comprehensive PyTorch Library for Efficient and Intelligent Deep Learning. `https://github.com/archinetai/smart-pytorch`*. Accessed on: March 15, 2024.*

Qiwei Bi, Jian Li, Lifeng Shang, Xin Jiang, Qun Liu, and Hanfang Yang. 2022. Mtrec: Multi-task learning over BERT for news recommendation. In Findings of the Association for Computational Linguistics: ACL 2022*, pages 2663–2669.*

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2020. 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.*

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. 2018. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993.*

Andriy Mulyar. 2022. Semantic Text Similarity Repository. `https://github.com/AndriyMulyar/semantic-text-similarity`.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. OpenAI blog*, 1(8):9.*

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune BERT for text classification? In Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings 18*, pages 194–206.*

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. Advances in Neural Information Processing Systems*, 33:5824–5836.*