SBRRT: Investigating Extensions on BERT

Stanford CS224N Default Project

Diego Valdez Duran
Department of Computer Science
Stanford University
diegoval@stanford.edu

Abstract

The robustness of models is limited due to the lack of extensive, high-quality data so training models on specialized tasks such as sentiment analysis, paraphrase detection, and semantic textual similarity is difficult. The nuanced nature of these tasks give rise to issues within training such as over-fitting and with generalizing to a broader context in the scope of their respective task. In this project, we aim to (1) re-implement a baseline miniBERT model, which generates robust contextual word representations, (2) investigate extensions and build upon upon a pre-trained miniBERT model using specific fine-tunings methods for multiple sentence-level tasks: utilizing a Smoothness-inducing regularization and Bregman proximal point optimization in fine-tuning using the framework developed by SMART [1](Jian et al.), a Round-robin multi-task learning approach to update all tasks simultaneously at a batch-level [2], and other fine-tuning in specific tasks such as Cosine-Similarity [3]. We explore efficiency and robustness in isolation and in combination with each of these techniques, from which we find that utilizing all these implementations in combination created the most robust model for these tasks; SMART mitigates over-fitting which was especially effective with a Round-robin multi-task learning approach.

1 Key Information to include

- Mentor: Gabriel Poesia | poesia@stanford.edu

2 Introduction

Transformers and large language models have revolutionized the field of natural language processing (NLP) in recent years. These models, built on deep learning architectures, have demonstrated remarkable success in a wide range of downstream NLP tasks, including language translation, question answering, summarizing, understanding textual information, among many others. The transformer architecture, introduced in 2017 by Vaswani et al., is now the de facto standard for many NLP applications due to its ability to model long-range dependencies and capture contextual information effectively.

Despite their impressive performance, large language models still face several limitations. One significant challenge is the huge computational cost required to build these models from scratch on specific downstream tasks, which are often very nuanced in their application making them inaccessible for many researchers and organizations. However, combining multi-task learning with transformer-based architectures, researchers have achieved significant improvements within these downstream NLP tasks. As an approach to address some of the limitations of large language models, Multi-task learning involves training a single model to perform multiple related tasks simultaneously, by sharing lower-level representations while allowing each task to have its own set of task-specific parameters. This approach has several advantages over training separate models for each task,

including improved generalization, reduced over-fitting, and better resource utilization. In fact, this is the core idea of this project and what we aim to investigate while incorporating specific fine-tuning techniques to improve upon a base miniBERT model.

More accurately, in this project we first finish the implementation of the original BERT model including multi-head self-attention and a Transformer layer. We then utilize the pre-trained embeddings from our default miniBERT implementation on sentiment analysis, paraphrase detection, and paired sentence semantic textual similarity. Before incorporating extensions to the model, we only perform sentiment analysis as a single-task classifier. Subsequently, we incorporate the extensions proposed in an effort to create a more robust and efficient miniBERT model: adopting a SMART fine-tuning technique, a Round-robin learning approach and utilizing other specific fine-tuning such as Cosine-Similarity. SMART regularization increased our performance by reducing over-fitting and improving generalization across domains by adding an additional loss term to the training objective of the BERT model, which encouraged the model to produce similar output for similar inputs across different domains. The specific multi-learning approach allowed us to update each of the tasks at a batch-level, creating more effective learning. In combination with a Round-robin learning approach, we found that we were able to significantly improve our baseline mniBERT results. We then explored possible modifications to the hyper-parameters of these techniques, but followed the standard hyper-parameters as directed by the SMART framework for implementation.

3 Related Work

While transfer learning has shown great promise in improving the performance of neural networks for natural language processing tasks, which is one of the core ideas on our tasks to other datasets, it also has some limitations. One of these limitations is that it requires large amounts of labeled data for pre-training, which may not be available for specific tasks or performances. The effectiveness of transfer learning also depends on the similarity between the pre-training task and the target task, and the availability of relevant pre-training data. If the pre-training task and data are not sufficiently similar to the target task, or if the amount of pre-training data is too limited, the transfer learning approach may not result in significant improvements in performance. Because these models are often overly dependent on the pre-training data and less adaptable to new data or domains, we address these issues in exploring techniques to mitigate over-fitting and create more robust adaptations to the model for the desired task.

We utilize SMART regularization specifically to address over-fitting, and we can use this in combination with other regularization techniques like dropout. Whereas dropout involves randomly dropping out nodes in the miniBERT model during training, preventing the model from relying too heavily on specific features, SMART regularization encourages the miniBERT model to learn shared representations that are relevant across the three different tasks, which can help reduce the impact of domain mismatch and improve overall quality of tasks. We followed the specific hyper-parameters that were laid out by SMART [Jiang et al], which allowed us to focus on the implementation rather than exploration of these fine-tunings, which was computationally and time heavy. In our search for optimization, we considered doing a grid-search or other hyper-parameter tunings in our model, but these came to be too time-consuming due to our constraints and resources (Google Co-lab).

We noticed that the original miniBERT implementation only contained fine-tuning on sentiment classification. Therefore, to create a more efficient and robust model we looked to address the limitations of our data by utilizing multi-task learning approaches where we aimed to train across the three downstream tasks simultaneously. More generally, multi-task learning can improve the performance of our BERT model of our desired tasks by leveraging shared information across them. There are a few different approaches that we considered or found, such as Hard parameter sharing, Task-specific layers, and Cross-stitch networks, but we found that a Round-robin learning approach seemed the most relevant to our project and goals. However, it was common to find that this created more of a risk of overfitting, due to the constraints of our datasets and so it was essential for us to consider regularization such as SMART as a way to mitigate this. However, through Round-robin we could leverage the task-specific data efficiently at a batch-level and find similarities between the three tasks while avoiding the need to store large amounts of data for each task separately. It also provides

a simple and effective way to balance the learning of each task, by ensuring that each task receives equal attention during training.

4 Approach

For this project, we utilize and build upon the the original BERTF model. Then, we extend on this model on the following downstream NLP tasks: sentence analysis (denoted SST), paraphrase detection (denoted PARA), and semantic textual similarity (denoted STS). We first utilize our completed miniBERT model, based on the original implementation, to only perform sentiment analysis on the Stanford Sentiment Treebank and CFIMDB datasets. In doing so, we create a baseline to evaluate our implementations on and to create an understanding on the level of performance of the original model. Also, we gain knowledge on how well this analysis generalized to the other tasks, which is a core idea of our project, exploring how we can improve these generalizations. We expand upon the complexities of our implementations below.

4.1 Round-Robin Multi-Learning Approach

We found that in the original miniBERT implementation there was only fine-tuning on sentiment classification, which was generalized to the other two NLP tasks. Therefore, to create a more efficient and robust model for generalization of these tasks, we implemented a Round-Robin multi-task learning approach to train across the three downstream tasks simultaneously. More specifically, once we loaded the pre-trained miniBERT weights, we iterated through the different datasets, including SST, Paraphrase, and STS, using a fixed batch size. During each iteration, updates for each dataset were made using their respective parameters. More specifically to each dataset, we utilized cross-entropy take make updates to sentiment (SST), binary cross-entropy for paraphrase (Quora), and mean-squared loss error loss for similarity (STS). Also, Cosine-Similarity between embeddings of a sentence pairs was used in semantic textual similarity (STS), which proved to be effective in experimentation as well.

Therefore, we were able to create updates at a batch level, which significantly improved our results. However, in doing so we also increased the risk for over-fitting. During experimentation, we were able to confirm this as our model achieved high results in training, but did not reflect similar results when evaluated on the dev sets. This motivated us to seek techniques to mitigate over-fitting and improve upon our model. We explore regularization and fine-tuning techniques below, using SMART as the main framework for the completion of our model.

4.2 Smoothness-Inducing Regularization

Within the process of fine-tuning, over-fitting has continually been a problem within models. Over-fitting can lead to poor performance in the test sets of the the three downstream NLP tasks of focus and generally on real-world applications because the model has essentially memorized the training data and is unable to make accurate predictions or generate coherent text for new inputs. In turn, it is critical that our fine-tuning efforts do not stray far from our pre-trained weights too fast. There are several techniques that can be used to prevent or mitigate over-fitting, but we choose to focus on regularization in parallel to SMART as a way to efficiently reduce over-fitting in our miniBERT model.

Following the framework from Jiang et al, we utulize SMART to effectively control the extremely high complexity of the model through Smoothness-inducing Adversarial Regularization. When there is a small perturbation to the input, this regularization ebciyrages for the output to change slowly (meaning a less drastic change). This creates a smoothness of the model, effectivley controlling the capacity of the model. In terms of our miniBERT model, we want to optimize the loss the following optimization for fine-tuning:

$$\min_\theta \mathcal{F}(\theta) = \mathcal{L}(\theta) + \lambda_s \mathcal{R}_s(\theta)$$

We then define the loss function as well as the smoothness-inducing adversarial regularizer, respectively below as laid out by Jiang et al:

$$\mathcal{L}(\theta) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(x_i;\theta), y_i)$$

$$\mathcal{R}_{\mathrm{s}}(\theta) = \frac{1}{n}\sum_{i=1}^{n} \max_{\|\widetilde{x}_i - x_i\|_p \leq \epsilon} \ell_{\mathrm{s}}(f(\widetilde{x}_i;\theta), f(x_i;\theta))$$

Continuing to follow this framework, we also adopt KL-divergence (laid out below). Here, classification tasks, $f(;\theta)$ outputs a probability simplex and ℓ is the symmetrized KL-divergence. However, for regression tasks $f(;\theta)$ results in a scalar and ℓ is a squared loss which is defined as: $(p-q)^2$. The smoothness-inducing adversarial regularizer involves a maximization problem, which is solved efficiently by projected gradient ascent. To elucidate exactly what this means, is that the regularizer is measuring the local Lipshitz continuity of f under the metric ℓ_s, which is stated in SMART. This is important as the smoothness created is helpful in mitigating over-fitting, and the output of f is encouraged to slowly change under injections of small perturbations well suited for domains that lack extensive resources.

4.3 Bregman Proximal Point Optimization

In combination with the regularization above, we also implemented a class of Bregman Proximal Point Optimization as the second proposition of SMART. This technique is helpful in preventing aggressive updating and the optimization method introduces a trust-region-type regularization (Conn et al., 2000) at each iteration. From this, updates to the model are only within a small neighborhood of the previous iterate. In turn, this effectively stabilizes the fine-tuning process by preventing aggressive updating. We provide the valilla Bregman proximal point method below, which we update parameters by at the $(t+1)$-th iteration:

$$\theta_{t+1} = \mathrm{argmin}_{\theta}\, \mathcal{F}(\theta) + \mu \mathcal{D}_{\mathrm{Breg}}(\theta, \widetilde{\theta}_t)$$

We then define Bregman Divergence as well as the regularized update step at the $(t+1)$-th iteration, respectively below as laid out by Jiang et al:

$$\mathcal{D}_{\mathrm{Breg}}(\theta, \theta_t) = \frac{1}{n}\sum_{i=1}^{n} \ell_{\mathrm{s}}(f(x_i;\theta), f(x_i;\theta_t))$$

$$\widetilde{\theta}_{t+1} \leftarrow (1-\beta)\bar{\theta}_S + \beta\widetilde{\theta}_t$$

Within the context of our own miniBERT implementation, we adapted the SMART framework by including the Smoothness Adversarial Regularization and Bregman Proximal Point optimization algorithms that Jiang et al presented. We were able to follow, for the most part, the original steps and logic behind these concepts. However, we also had to make adaptations to allow these techniques to be applicable to the miniBERT model which utilized dropout techniques as well. It's important to note that in our results section, we find diminishing returns in outputs for sentiment analysis. This could have been due to these complications, as calculations such as KL-divergence loss is affected. In our miniBERT model, we kept dropout layers within the Multi-class implementation, which could have hindered our results in combination with SMART.

5 Experiments

5.1 Data

In terms of original miniBERT model, it was trained on two unsupervised tasks on Wikipedia articles: masked language modeling and next sentence prediction. For fine-tuning of the model, all the following datasets were pre-processed by tokenizing sentences strings, lower-casing said tokens, standardizing punctuation tokens, and padding sentences to allow for matrix multiplication. We utilized the Stanford Sentiment TreeBank (SST) dataset, consisting of $11,855$ single sentences from movie reviews extracted from movie reviews. Each phrase contains a negative, somewhat negative, neutral, somewhat positive or positive label. This dataset is split accordingly: train (8,544 examples), dev (1,101 examples) and test (2,210 examples). The Quora dataset consisted of 400,000 questions

pairs containing binary labels, simply true or false depending if a text was a paraphrase of the other (similar sentiment portrayed). The CFIMDB dataset was also utilized, containing $2,434$ highly polar movie reviews. Each movie review contains binary labels of negative or positive. Many of these reviews are longer than one sentence. This dataset is split accordingly: train (1,701 examples), dev (245 examples) and test (488 examples).

5.2 Evaluation method

In terms of evaluation, we mainly use an accuracy (SST, Quora) to quantify sentiment analysis and a Pearson score for semantic textual similarity (SemEval). We know that paraphrase and CFIMDB labels are binary, while semantic textual similarity is multi-class classification. We also use iterations per second as a metric to measure run-time for each of our models, which can give us insights on the computational costs and complexities of each model.

5.3 Experimental details

In all models, we run 10 epochs for each implementation to ensure that we train models equally and sufficiently. With our default pre-train and fine-tune, we ran our preliminary tests on a learning rate of 1e-3 and 1e-5 respectively. We also utilized the original dropout probability of 0.1. As per the original implementation, we use a batch size of 64 for sentiment analysis and 8 on CFIMDB.

SMART introduces many other hyper-parameters that can be tweaked, however due to our computational and time constraints we follow the recommendations by the original papers [Jiang et al.]. For Smoothness Adversarial Regularization, we have $\lambda = 5, \epsilon = 1e-5, \sigma = 1e-5, \beta = 0.95, \mu = 0.8$. We adjusted β and μ for our experimentations so these values differ from the original, however we were unable to find optimal tunings for our BERT implementation due to limitations.

For our Round-robin approach, we set the batch size to 8 as to accompany a lower learning rate from before. In the original implementation, this value is set to 64 but when utilizing SMART we use 8. To calculate the number of iterations, we divide the number of training data from STS by the batch size imputed for each of the tasks for equality.

5.4 Results

Based on our table below, we see the performance of our model based on pre-training and fine-tuning of our default model. We compare dev accuracy to the benchmarks given in the default project handout, where we see that we surpass or reach similar level-performance to the benchmarks. As a result, we were confident in our default implementation. We then gained some insight on how we could improve our model using the table above as foundation.

Prediction Task	**Sentiment Classification (SST)**		**Sentiment Classification (CFIMDB)**	
Model	Implemented	Benchmark	Implemented	Benchmark
Default Pre-train	0.416	0.390	0.780	0.780
Defaut Fine-tune	0.531	0.515	0.963	0.966

Figure 1: Comparison of baseline ac curacies on initial pre-trained and fine-tuned BERT model for sentiment calssification on a single-task classifier in a Google Colab environment

In our second table, we explore the performances of our implementations in isolation and combination with one another. Interestingly, we are unable to surpass our baseline results for sentiment classification. This could be due to issues regarding the implementation of this task. There may have been increased performance for sentiment if we instead gave less weight to this task, but due to our limitations we proceeded with our proposed methods without additional exploration of this. Looking at the table above, we find that our proposed methods do improve our baseline miniBERT,

Extended Model	Sentiment Classif.	Paraphrase Detection	Semantic Similarity	Iterations / sec (average)
Default: BERT (pretrain)	0.415	0.609	-0.072	–
Default: BERT (fine-tune)	0.529	0.381	0.246	8.09it/s
Extension: SMART	0.518	0.371	0.307	3.36it/s
Extension: Round-Robin Learning	0.500	0.695	0.496	2.20it/s
Extension: SBRRT	0.505	0.719	0.567	1.45it/s

Figure 1: Comparison of scores across each implementation on default BERT model in isolation as well as in combination with one another on the SST, Quora, and STS datasets/tasks

but in combination of our methods we are able to achieve the best result. We see that for paraphrase detection and semantic textual similarity, we have the greatest scores by using a combination of SMART as well as the Round-robin multi-task learning approach. However, we note that we have diminishing returns for sentiment classification.

We have highlighted our best scores in the table and the following are the best for each task: 0.529 (SST), 0.719 (Quora), 0.567 (STS). We have also listed the number of iterations per second on each model, as a way to compare run-times across all our implementations. Not surprisingly, we see that more simple models achieve lower computational costs and time. On the other hand, the most complex model, a combination of SMART and Round-robin which we denote as SBRRT, is more computationally heavy and takes more time as we have less iterations per second.

6 Analysis

In our analysis, we will go in depth on what succeeded and failed within our attempts in implementation of the miniBERT model. We found that by fine-tuning the baseline miniBERT model we were able to improve performance on the original baseline task (SST). However, the original model assumed that this would generalize to the other two NLP downstream tasks, which resulted in poor performance. We can see this within our results of our baseline implementation, where we saw a single-task approach being insufficient in testing. In turn, this lead to the consideration and implementation of a multi-learning approach as a way to leverage our resources effectively. By utilizing a Round-robin multi-task learning approach, we were able to significantly improve upon our baseline model. In combination with this, we also utilized regularization techniques laid out by SMART as a way to mitigate over-fitting and create a more robust model able to generalize across all three sentence level tasks efficiently.

By utilizing SMART, we are able to use a regularization technique called encourage the miniBERT to learn shared features across tasks. This helps to prevent the model from over-fitting to a specific task and instead learn more generalizable representations. We saw an improvement of around 0.61 in correlation with the implementation of SMART. However, we also saw diminishing returns of about 0.1 percent in accuracy across sentiment analysis as well as paraphrasing in this implementation. When using a Round-robin multi-task learning approach, we saw a 0.314 increase in accuracy in paraphrase compared to our default fine-tuned model and a 0.25 increase in correlation in semantic textual similarity. More significantly, we saw the greatest results when combining both approaches

were we had a high of 0.719 and 0.567 in paraphrase detection and semantic textual similarity when using SMART and Round-robin together (SBRRT). While we did see diminishing returns in semantic analysis, this still proceed to be the most effective model across our sentence-level tasks as a whole.

7 Conclusion

In this project, we extended and build upon upon a pre-trained miniBERT model using specific fine-tunings methods for sentiment analysis, paraphrase detection and semantic textual similarity by utilizing a Smoothness-inducing regularization and Bregman proximal point optimization in fine-tuning, a Round-robin multi-task learning approach to update all tasks simultaneously at a batch-level [2], and other fine-tuning in specific tasks such as Cosine-Similarity [3]. We found that utilizing all these techniques in combination created the most robust model for these tasks; SMART mitigates over-fitting which was especially effective with a Round-robin multi-task learning approach. Our highest performing model achieved SST test Accuracy: 0.505, Paraphrase test Accuracy: 0.721, STS test Correlation: 0.536 and and Overall test score: 0.587 which was around number 80 on the leader-board before the first deadline.

However, we found diminishing results within sentiment specifically within extensions to the miniBERT model. This may have been caused by the weights assigned to this task, and could have been mitigated by adjustments. Due to the time and computational constraints of the project, we would have liked to explore hyper-parameter tunings through techniques like grid-search for optimal performances as well as utilizing additional pre-training, which was a goal in the preliminary stages but was not implemented due to constraints.

References

[1] Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. Smart: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. arXiv preprint arXiv:1911.03437, 2019.

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

[3] CS 224N: Default Final Project: minBERT and Downstream Tasks

[4] Nils Reimers and Iryna 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), pages 3982–3992, 2019

[5] Nils Reimers and Iryna Gurevych. 2018. Why Comparing Single Performance Scores Does Not Allow to Draw Conclusions About Machine Learning Approaches. arXiv preprint arXiv:1803.09578, abs/1803.09578.

[6] Johannes Fürnkranz. 2002. Round robin classification. J. Mach. Learn. Res. 2 (3/1/2002), 721–747. https://doi.org/10.1162/153244302320884605

[7] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 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. Springer, 2019.

[8] Nils Reimers and Iryna 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), pages 3982–3992, 2019.

[9] Antti Tarvainen and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In Advances in neural information processing systems, pages 1195–1204.

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

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