Fine-tune miniBERT for multi-task learning

Stanford CS224N Default Project

Geo Zhang
Department of Energy Science and Engineering
Stanford University
`gmzhang@stanford.edu`

Abstract

This project investigates the effectiveness of fine-tuning BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2018), a pre-trained model, on multi-task learning. We employ miniBERT, which comprises 12 layers and a total of 110 million parameters, to simultaneously address three distinct linguistic challenges: sentiment analysis, paraphrase detection, and semantic textual similarity. The training and evaluation are based on the Stanford Sentiment Treebank (SST), the Quora Dataset (Para), and the SemEval STS Benchmark (STS), respectively. Accuracy metrics are utilized for sentiment analysis and paraphrase detection, while the Pearson correlation coefficient assesses semantic textual similarity. Our experimental approach includes building a sentence-BERT (Reimers and Gurevych, 2019) adaptation with different loss functions and sentence-concatenate multitask BERT with different fine-tuning approaches. Sentence-concatenating helps BERT model extract the inner correlation between sentences, contributing to the performance on the Para and STS tasks. Both sequential and parallel fine-tuning of three tasks are explored. We adopted the sequential strategy with sentence-concatenate as our baseline. Furthermore, we experiment with the complexity of task-specific heads, incorporating additional layers and projected attention layers (PALs) Stickland and Murray (2019)), as well as employing additional regularization. Our findings indicate that a sentence-concatenation structure with task-specific heads and epoch-wise learning rate decay significantly enhances performance. This study not only underscores the versatility of miniBERT in multitask learning but also advances our understanding of fine-tuning strategies for downstream tasks.

1 Key Information to include

- Mentor: Timonthy Dai
- External Collaborators (if you have any): N/A
- Sharing project: N/A

2 Introduction

One of the main challenges in deep learning is the high data requirements and computational costs. Transfer learning, which involves fine-tuning a pretrained model for specific tasks, provides an effective solution. However, when dealing with multiple tasks, a key decision is whether to allocate a separate model to each task or to use a shared model for various tasks. Multitask learning (MTL) is particularly beneficial in this context. For example, in situations with limited storage on lightweight devices, a single multipurpose model can efficiently manage multiple tasks at minimal cost. Additionally, MTL closely resembles human learning processes, where individuals build upon existing knowledge and skills rather than starting from scratch. This makes MTL a more natural and efficient approach to learning.

In this project, we utilize the BERT model (Devlin et al., 2018), a pre-trained language model from the transformers architecture known for generating high-quality context embeddings based on extensive pre-training on large corpus. BERT's simplicity and effectiveness have made it popular for various natural language processing (NLP) tasks, including text classification (Sun et al., 2019), although initial studies predominantly focused on single-task applications. Our investigation aims to enhance BERT's embeddings across multiple tasks through multi-task fine-tuning. We focus on three tasks: sentiment analysis, paraphrase detection, and semantic textual similarity. The project is structured into several key components: (1) We implement a Sentence-BERT architecture with multiple loss functions to optimize MTL performance. (2) We explore sentence-concatenation approaches and experiment with sequential and parallel multitask fine-tuning strategies. (3) We introduce complexity into the task-specific heads and implement learning rate decay to enhance performance.

3 Related Work

BERT has achieved remarkable results across a range of downstream NLP tasks. In the original paper by Devlin et al. (2018), BERT was fine-tuned for 11 different NLP tasks, including those in the General Language Understanding Evaluation (GLUE) benchmark. Sun et al. (2019) provided a comprehensive overview of BERT fine-tuning methodologies and strategies for various text classification tasks. Zhang et al. (2019) explored several fine-tuning techniques in few-sample scenarios. When it comes to fine-tune multi-tasks, which is a relatively underexplored aspect of BERT fine-tuning. Zhang and Yang (2021) and Crawshaw (2020) offer insights into MTL, discussing architectures, optimization methods, and task relationship learning in this context. Sentence-BERT, introduced by Reimers and Gurevych (2019), employed a Siamese network architecture with two BERT encoders sharing weights and custom loss function designs for sentence-pair inputs, making it well-suited for handling many sentence input scenarios.

4 Approach

4.1 The sentence-BERT model

For the first sentiment analysis task, we extract the sentence embedding from the pre-trained BERT model, using only the [CLS] token, and connect it to a task-specific head to determine the appropriate sentiment score. This structure is relatively simple to implement. For the input of sentence pairs for comparison, following a similar scheme, we use a Sentence-BERT (SBERT) model (Reimers and Gurevych, 2019), which is designed to produce semantically meaningful sentence embeddings for similarity comparison. The structure is illustrated in Fig. 1. The input consists of two sentences, which generate a pair of sentence embeddings through identical BERT encoders. We compare these embeddings using similarity measures such as cosine similarity, and employ a loss function to quantify and fine-tune the model.

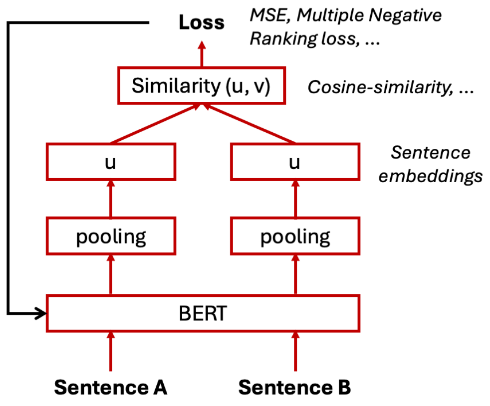

Figure 1: Sentence-BERT architecture for sentence pair inputs; where the same BERT are used to generate sentence embeddings.

Cosine-similarity is a very common measure for vector embeddings. The cosine-similarity between the two sentence embeddings u and v is computed as,

$$\text{COSINE-SIM}\left(u, v\right) = \frac{u \cdot v}{\|u\|_2 \|v\|_2}. \tag{1}$$

comparing it with the similarity score provided with data set, which ranges from 0 to 5, we scale up the COSINE-SIM (u, v) by 5 and using mean-squared-error loss as the objective function suggested by Reimers and Gurevych (2019),

$$L\left(u, v, \hat{y}\right) = \frac{1}{n}\Sigma_{i=1}^{n}\left(\hat{y}_i - \text{COSINE-SIM}\left(u_i, v_i\right)\right)^2. \tag{2}$$

To enhance the embedding generation performance of BERT in the paraphrase detection task, we adopted the Multiple Negative Ranking (MNR) loss function introduced by (Henderson et al., 2017). This loss function is designed for datasets in the form (u_i, v_j), where $i = j$ are similar pairs and $i \neq j$ are dissimilar pairs. We made slight modifications to adapt it for our paraphrase detection task, using samples with the true label $\hat{y}_i = 1$,

$$L\left(u, v\right) = -\Sigma_{i=1}^{n} \log \frac{e^{\text{SIM}(u_i, v_i)}}{\Sigma_j^n e^{\text{SIM}(u_i, v_j)}}, \tag{3}$$

In this context, the similarity between paraphrase pairs u_i and v_i should be maximized, while the similarity between non-paraphrase pairs u_i and v_j will be minimized.

4.2 Multitask fine-tuning scheme

Instead of the Sentence-BERT approach, we also explore another scheme illustrated in Figure 2. We believe that concatenating a pair of sentences and feeding them into BERT for similarity tasks can enhance the model's understanding of the relationship between the two sentences. When sentences are concatenated, BERT's attention mechanism can then focus on relevant parts of both sentences simultaneously, allowing words or phrases in one sentence to directly influence the representation of words or phrases in the other sentence. This leads to a more accurate similarity assessment.

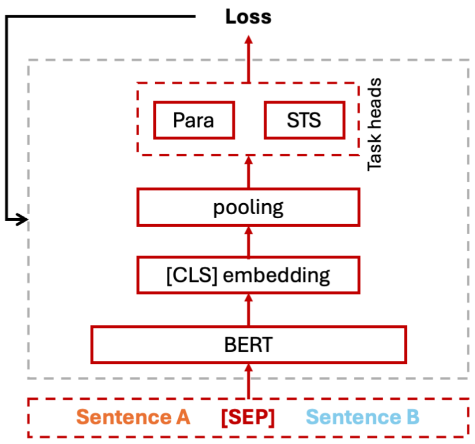

Figure 2: Concatenate a pair of sentences and input into BERT for similarity analysis.

In the MTL fine-tuning process, one question is how to arrange the order of different tasks. Here we probe the two routines: training sequentially and parallelly. An illustration of these routines is shown in Fig. 3.

Sequential Training Routine: In the sequential training routine, we switch between tasks at the end of each sub-epoch. An epoch consists of multiple sub-epochs, each dedicated to a different task.

Parallel Training Routine: In the parallel training routine, we switch between tasks at the end of each batch, recording the total loss for three different tasks. Within one epoch, we train all three tasks simultaneously without focusing the model specifically on any single task.

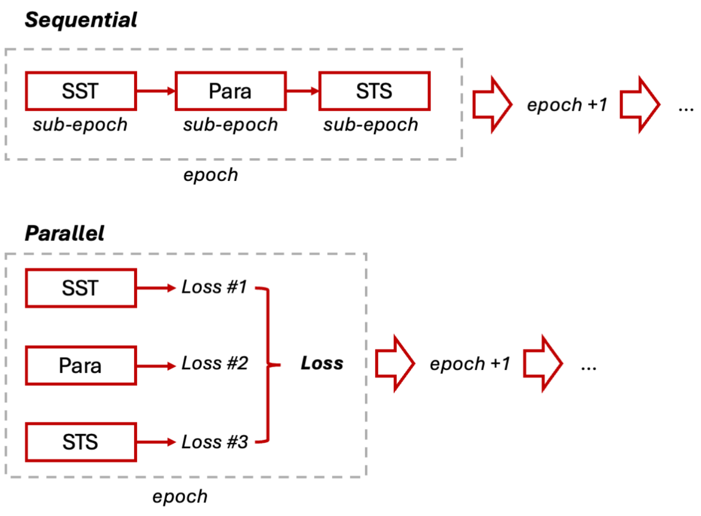

Figure 3: Sequential or parallel routines of fine-tuning BERT for MTL

4.3 Task-specific Adaptions

The original idea was proposed due to the fine-tuning process involving many parameters, if we could freeze the transferred BERT layers and only fine-tune the lastly added task-specific layers, that would save a lot of resources (Stickland and Murray, 2019). Also, due to the inherent differences between tasks, using task-specific heads to adapt to different tasks can significantly enhance multi-task learning performance. In this case, We experimented with two simple schemes to evaluate their effectiveness.

In the first scheme, we added a simple linear layer on top of the BERT model. This layer acts as a task-specific head, providing a direct mapping from the BERT embeddings to the output space of each task. The simplicity of this approach ensures minimal computational overhead and facilitates faster training. We further experimented by adding an additional linear layer followed by a normalization layer to see whether a more complex head would lead to better results.

In the second scheme, we explored the use of a projected attention layer on top of BERT as Stickland and Murray (2019). Different tasks often require different representations and mappings. Task-specific heads allow the model to tailor its output layer to the unique requirements of each task, improving overall performance.

By employing these schemes, we aim to determine whether a more complex task-specific head can better interpret the embeddings produced by BERT, ultimately leading to improved performance in MTL.

5 Experiments

5.1 Data

The datasets employed for this project are the default project provided as shown in 1. Owing to the extensive size of the Quora dataset, we initially reduced the training subset to one-third of its original size at random to facilitate the test of various fine-tuning methodologies. Subsequently, for the generation of final results, the complete dataset was utilized.

5.2 Evaluation method

Our evaluation methods adhere to the guidelines outlined in the default project handout. For the sentiment analysis (SST) and paraphrase detection (Para) tasks, we employ classification accuracy as the evaluation metric, defined as the ratio of correct predictions to the total number of examples: $accuracy = \frac{true\ predictions}{number\ of\ examples}$. For the semantic textual similarity (STS) task, we utilize the

Table 1: Provided datasets for the default projects with three downstream tasks.

Task	Datasets	Size	Labels
Sentimental analysis	Stanford Sentiment Treebank (SST)	Train: 8,544 examples Dev: 1,101 examples Test: 2,210 examples	Movie reviews with 0, 1, ..., 5 categorical labels
Paraphrase detection	Quora Dataset	Train: 283,010 examples Dev: 40,429 examples Test: 80,859 examples	Sentence pairs labeled as True (1) or False (0)
Sentence similarity	SemEval STS Benchmark Dataset	Train: 6,040 examples Dev: 863 examples Test: 1,725 examples	Similarity score from 0 to 5

Pearson Correlation Coefficient (PCC) to measure the correlation between the actual similarity scores and the predicted ones. The equation writes as,

$$r\left(y, \hat{y}\right) = \frac{\sum(y_i - \bar{y})(\hat{y}_i - \bar{\hat{y}})}{\sqrt{\sum(y_i - \bar{y})^2 \sum(\hat{y}_i - \bar{\hat{y}})^2}} \tag{4}$$

where r ranges from -1 to 1, y_i indicates predicted label while $\hat{y}$ is the true score.

5.3 Experimental details

For all our experiments, we fine-tuned the entire model (BERT along with additional components tailored for specific tasks) using a learning rate of 1×10^{-5}. Each model was trained over 10 epochs with a default batch size of 8. To save the training cost for the paraphrase detection task, we employed a reduced training set, as previously described, to evaluate the effectiveness of different fine-tuning methods. The comparison revealed that the results from the smaller Quora dataset closely approximated those from the full dataset, with single-task accuracies of 0.857 and 0.870, respectively.

Initially, each task was equipped with a task-specific head atop BERT, receiving the hidden state of the [CLS] token as input. In our first experiment, we implemented a sentence-BERT framework using cosine-similarity mean squared error (MSE) and multiple negative ranking (MNR) loss functions. This configuration allowed for the input of two sentences into BERT to produce separate embeddings, which were then compared directly. We trained the tasks sequentially within each epoch.

Subsequently, we shifted from the sentence-BERT method to a concatenation approach, where sentences were combined prior to input into BERT. This produced a single embedding from which the relationship between the two sentences was inferred using the [CLS] token's hidden state. The loss function is simply mean-square-error and AdamW optimizer is adopted. Furthermore, we assessed the performance impact of training tasks in a sequential versus parallel arrangement, details of which are elaborated in the Approach section. We also explored increasing the complexity of the task-specific heads by adding an additional hidden layer with 256 units, implementing normalization, and integrating a projected attention layer. Ultimately, an epoch-wise learning rate decay strategy—halving the learning rate each epoch is tested on different models.

5.4 Results

The performance scores on our development dataset are detailed in Table 2. We explore the multi-task fine-tuning routines discussed in Section 5.3 and compare these results against both our baseline and single-task fine-tuning methods. The baseline involves using a BERT model fine-tuned sequentially on tasks with concatenated sentences.

Our observations indicate that the sequential multi-task fine-tuning routine can produce embeddings that lead to performance on par with that of single-task fine-tuning across each downstream task, as well as with the parallel multi-task fine-tuning approach. Additionally, we note varying impacts of the increased complexity in task-specific heads across different datasets within the multi-task framework. The role of the learning rate also proves crucial in influencing the outcomes.

Our best results were achieved using the enhanced base model, by adding an additional hidden layer to the SST-head and implementing a learning rate decay strategy during training on the complete Quora dataset. The outcomes on the test leaderboard are detailed in Table 3. A comprehensive analysis of these results is provided in the subsequent Section 6.

Table 2: Dev accuracy/Pearson correlation results with different experiments.

Method	Dev. SST	Dev. Para	Dev. STS
multitask BERT + Cos-Sim for STS + MNR for Para	0.480	0.666	0.740
BERT + concat sentences (single task performance)	0.510	0.857	0.858
Multitask parallel + concat sentences	0.502	0.850	0.869
Multitask sequential + concat sentences (base model)	0.513	0.850	**0.875**
Base model + additional hidden layer (256)	0.498	0.839	0.873
Base model + on-top Projected Attention Layer	0.488	0.841	0.869
Base model + lr decay (0.5)	0.517	**0.857**	0.867
Base model + hidden + lr decay (0.5)	**0.520**	0.855	0.865
Base model + hidden layer on SST + lr decay + full Quora	0.520	0.885	0.867

Table 3: Final results on the test set (leaderboard)

	Test SST	Test Paraphase	Test STS	Test mean
Score	0.518	0.886	0.874	0.780

6 Analysis

Initially, a sentence-BERT methodology incorporating cosine-similarity MSE loss and MNR loss was employed, with the outcomes displayed in the first row of the table. A significant enhancement was achieved by transitioning from sentence-BERT to a sentence concatenation approach. This change notably improved the accuracy of the paraphrase (Para) detection task from 0.666 to 0.857 and increased the Pearson correlation coefficient for the semantic textual similarity (STS) task from 0.740 to 0.858. Rather than merely comparing the embeddings of the two sentences, concatenating them—with a [SEP] token inserted between—allowed BERT to leverage the contextual information of the preceding sentence to better encode the following one. Subsequently, we integrated the three tasks (sentiment analysis (SST), Para, and STS) into a sequential multitask framework, which served as the base model. The training for each task was carried out sequentially within a sub-epoch, starting with the sentiment analysis (SST) task, followed by the paraphrase (Para) detection task, and concluding with the semantic textual similarity (STS) task. This sequence constituted one full epoch before the models were evaluated on the development set. This strategy enhanced the performance of the SST task, with accuracy improving slightly from 0.510 to 0.513, and more noticeably for the STS task, from 0.858 to 0.875. However, it detrimentally affected the performance of the Para task. These results suggest that multitask training can have mixed effects, where the tasks may either positively or negatively influence each other's performance.

From our experiments with multi-task fine-tuning routines, we observed that the sequential multi-task fine-tuning approach produced robust embeddings, yielding performance comparable to that of parallel multi-task fine-tuning for each tasks. Contrary to our initial expectations, the knowledge learned from tasks trained earlier did not diminish during the training of subsequent tasks as anticipated. Initially, we suspected that earlier task knowledge would be forgotten as later tasks were learned. This is maybe due to training multi-tasks parallel will introduce a new issue: the combined loss from all tasks was not sufficiently sensitive to improvements in any single task.

We also investigated whether employing a more complex task-specific head layer could better capture underlying relationships. Surprisingly, very complex heads, such as Projected Attention Layers (PALs), did not enhance performance. In contrast, the addition of a simple linear layer followed by normalization significantly improved performance on the SST task. This suggests that capturing the nuances of sentiment requires a more sophisticated interpretation of the [CLS] token's hidden state, which was effectively provided by this simpler modification.

The learning rate plays a crucial role in model performance; a large learning rate can cause drastic parameter changes, leading to oscillating results that fail to improve. Implementing a decaying learning rate strategy helps to fine-tune the adjustments to the parameters more gradually, facilitating the achievement of optimal performance. This approach ensures that the learning rate decreases over time, preventing the model from overshooting the minimum of the loss function and promoting a more stable convergence.

There were also several experiments conducted that are not detailed in the results. One such experiment involved the implementation of the Sandwich Layer Norm, as introduced by Ding et al. (2021). This technique adds additional normalization both before and after the multi-head attention layer and the feed-forward layer. Contrary to expectations, this modification actually deteriorated the performance in our tests. We believe that the reason for this decline is the incompatibility between the pretrained model parameters and the modified structure of BERT. This suggests that even minor structural changes, such as the addition of a normalization layer that does not introduce new parameters, can significantly impact the pretrained model's effectiveness.

7 Conclusion

In this project, our fine-tuning of BERT for MTL revealed several key findings:

Sentence Concatenation significantly improved paraphrase detection (Para) and semantic textual similarity (STS) tasks by enabling BERT to encode richer contextual relationships between sentences. Sequential Multitask Training, contrary to our expectations, did not lead to forgetting earlier tasks. Task-specific head layers needs to be designed specifically, a complex task-spefiic head like PAL did not guarantee enhanced performance. Instead, a simple linear layer with normalization significantly improved the score on SST task, indicating that a simpler modification could be more effective. A decaying learning rate strategy was essential for optimal performance. A large learning rate can cause overshoot, while a gradually decreasing rate facilitates stable convergence.

Despite these findings, there are many unexplored fine-tuning approaches for improving BERT's performance on MTL. In the future, we plan to investigate different techniques to identify those that generally enhance BERT's performance.

References

Michael Crawshaw. 2020. Multi-task learning with deep neural networks: A survey. *arXiv preprint arXiv:2009.09796*.

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*.

Ming Ding, Zhuoyi Yang, Wenyi Hong, Wendi Zheng, Chang Zhou, Da Yin, Junyang Lin, Xu Zou, Zhou Shao, Hongxia Yang, et al. 2021. Cogview: Mastering text-to-image generation via transformers. *Advances in Neural Information Processing Systems*, 34:19822–19835.

Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. *arXiv preprint arXiv:1705.00652*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Asa Cooper Stickland and Iain Murray. 2019. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning. In *International Conference on Machine Learning*, pages 5986–5995. PMLR.

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. Springer.

Yu Zhang and Qiang Yang. 2021. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*, 34(12):5586–5609.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. Ernie: Enhanced language representation with informative entities. *arXiv preprint arXiv:1905.07129*.