# BERT-MTS: Fine Tuning BERT for Multi-Task Serving

Stanford CS224N Default Project

**Name**
Department of Computer Science
Stanford University
nkanakia@stanford.edu

## Abstract

The project has a two fold goal: implement core components of BERT to better understand its architecture, and experiment with possible extensions to produce generalized sentence embeddings that can perform well on multiple tasks. For the first goal, important components of BERT were implemented to create a minBERT model. As for the second goal, three task specific minBERT models were fully fine-tuned for comparison with a single multi-task model built using the framework proposed in Wei et al. (2022). The fully fine-tuned task specific models outperformed the pre-trained minBERT model, and the single multi-task model achieved $99\%$ accuracy of the fully fine-tuned models.

## 1 Key Information to include

- Mentor: Gabriel Poesia Reis e Silva
- External Collaborators (if you have any): N/A
- Sharing project: N/A

## 2 Introduction

While the first goal of the project is interesting as it is a learning exercise to better understand the BERT architecture, the second one is more exciting due to its open ended nature. The second goal is challenging in that it is difficult to produce robust model embeddings that perform well across several tasks. Problems such as forgetfulness and conflicting gradient directions plaque the performance of models in a multi-task setting. There are myriad approaches that can be experimented to produce generalized BERT embeddings. Utilizing the framework proposed in Wei et al. (2022) to serve BERT for multiple tasks provides the ability to independently fine-tune and update the model for frequently changing tasks without affecting other tasks. It enables pre-trained BERT embeddings to be shared across multiple tasks. The key enabling idea is that only some of the top layers of BERT are fine-tuned specific to the task, and the bottom layers are kept frozen to be shared across them. The work for the second part of the project evaluates the framework on minBERT for the task of sentiment analysis (SA), semantic textual similarity (STS) and paraphrase detection (PD), and confirms that partial fine-tuning of the top most layers produces performance close to that of full fine-tuned minBERT models on the aforementioned tasks. In addition, we also evaluate how cosine similarity (CS) with the mean-square-error (MSE) loss objective perform in comparison to a linear head with cross entropy (CE) loss on the task of STS.

## 3 Related Work

The partial fine-tuning approach has been studied before in ((Houlsby et al., 2019); (Merchant et al., 2020)) and is a sensible idea given that previous studies show that the middle layers of BERT are

most transferable, and the top layer representations are task oriented. ((Wang et al., 2019); (Tenney et al., 2019); (Liu et al., 2019); (Merchant et al., 2020)]). (Merchant et al., 2020) demonstrated that fine-tuning primarily affects weights from the top layers while weights in the bottom layers do not change much. The use of CS and MSE loss has been suggested in Reimers and Gurevych (2019) as a regression objective for a siamese setup of the BERT model. A similar approach is adopted in this work and compared with the CE loss objective on the task of STS.

# 4 Approach

## 4.1 Main Approach

The following steps are performed for the second part of the project, as depicted in Fig. 1.

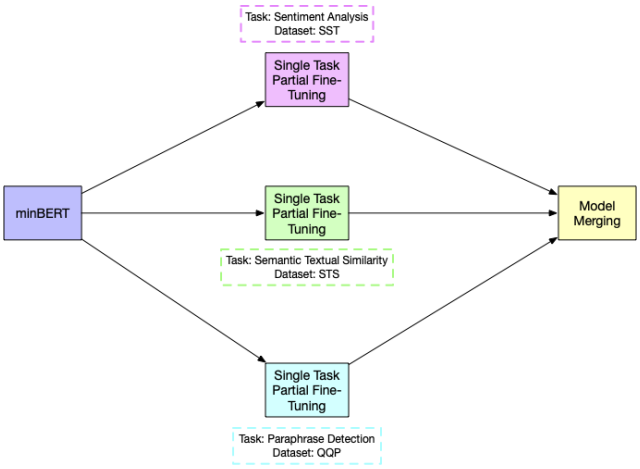


Figure 1: minBERT serving pipeline

### 4.1.1 Single Task Partial Fine-Tuning

This step involves fine-tuning an independent copy of BERT for each of the three tasks while keeping some layers frozen. This steps produces a set of single-task models referred to as single-task teacher models. The CE loss objective was used as a baseline to fine-tune the model.

### 4.1.2 Single Task Knowledge Distillation (KD)

KD (Buciluundefined et al., 2006) is a compression technique in which a compact model, the student, is trained to reproduce the behavior of a larger model, the teacher. The goal of this step is to compress the fine-tuned task-specific layers in the teacher model to smaller number of layers in a student model. The framework in Wei et al. (2022) proposes this step when serving the model under a resource constrained environment. Since we do not have such a constraint we skip this step for the work in this project.

### 4.1.3 Model Merging

The final step involves merging the fine-tuned single-task models into a multi-task model such that the parameters and computations in the frozen layers can be shared. This is achieved by loading weights from multiple model checkpoints into a single computation graph, as depicted in Fig. 2.

## 4.2 STS Loss Objectives

STS specific full fine-tuning of minBERT is conducted on two separate loss objectives and the best performing is chosen for further evaluation. In the baseline approach sentence pairs are concatenated with the [SEP] token and fed as an input to the minBERT model. The token type ids that were a placeholder in the base minBERT model are now considered for constructing the input embeddings.

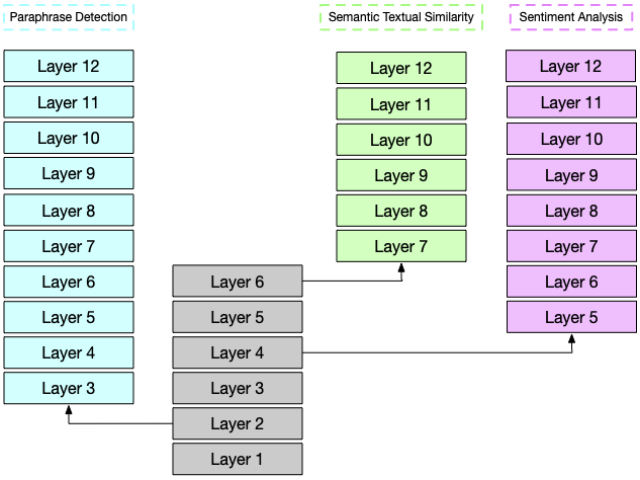


Figure 2: minBERT single task model merging

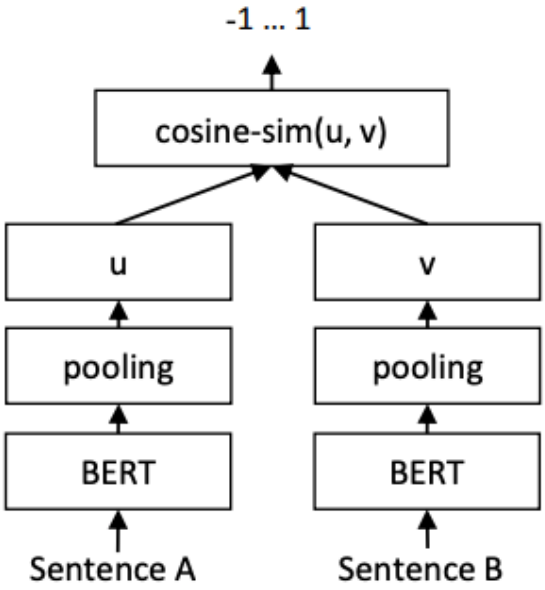


Figure 3: SBERT architecture (Reimers and Gurevych, 2019)

A linear head is used to generate logits from the pooled output and loss is calculated using the CE objective.

The baseline is compared with a saimese setting of the model in which the individual sentences of a pair are fed into the same model separately, as depicted in Fig. 3 (Reimers and Gurevych, 2019). A cosine similarity between the pooled outputs is computed and MSE is used as the loss objective for fine-tuning. The pooled out for all procedures in this project is the [CLS] token embedding from the last minBERT layer.

## 5 Experiments

### 5.1 Data

Stanford Sentiment Treebank Socher et al. (2013) for SA, SemEval Agirre et al. (2013) for STS and Quora Question Pairs Chen et al. (2017) for PD.

- Stanford Sentiment Treebank (SST) dataset
  - train (8,544 examples)
  - dev (1,101 examples)
- CFIMDB dataset
  - train (1,701 examples)
  - dev (245 examples)
- Quora Dataset
  - train (141,506 examples)
  - dev (20,215 examples)
- SemEval STS Benchmark Dataset
  - train (6,041 examples)
  - dev (864 examples)

### 5.2 Evaluation method

Accuracy percentage is used to evaluate model performance for the task of SA and PD, while Pearson Correlation is used for STS.

### 5.3 Experimental details

The following minBERT models were evaluated independently for each of the respective tasks on task-specific dev data sets.

- Base model (baseline)
- Single task fully fine-tuned models
- Partially fine-tuned merged model

The models were fine-tuned on task-specific training data sets with the following configurations:

- learning rate: $2e-5$
- Adam: $\beta 1 = 0.9, \beta 2 = 0.999$
- epochs: $3, 4, 5$
- dropout: $0.3$
- batch size: $64$
- top most layers: 4, 5, 6, 7, 8, 9, 10 (Wei et al., 2022)

### 5.4 Results

Most of the results are inline with expectations, wherein fine-tuning boosts performance and partial fine-tuning of the top most layers produces performance close to fully fine-tuned models. Results are shown in Table 1, Table 2, Table 3 and Table 4.

## 6 Analysis

Better performance results were expected on the task of STS with the use of CS and MSE loss. The poorer than expected performance could be due to the following reasons.

- The default implementation of the MSE loss objective penalizes the outcome even when CS results in a negative value $[-1, 0)$ and the sentences are dissimilar. It forces the CS output to be 0, i.e. the sentence embeddings to be orthogonal, even when it would suffice if they are anywhere in between orthogonal and opposite.
- The labels are normalized to the $[0, 1]$ interval during training and evaluation. This does not map neatly to the cosine interval of $[-1, 1]$, resulting in a misaligned loss objective.

| minBERT top layers/Task | SA | STS | PD |
|---|---|---|---|
| layer 4 | 0.5 | 0.818 | 0.882 |
| layer 5 | 0.521 | 0.830 | 0.884 |
| layer 6 | 0.504 | 0.802 | **0.887** |
| layer 7 | **0.526** | 0.802 | 0.879 |
| layer 8 | 0.510 | 0.839 | **0.887** |
| layer 9 | 0.510 | 0.835 | **0.887** |
| layer 10 | 0.515 | **0.849** | **0.887** |

Table 1: dev accuracy of minBERT task-specific layers: (4, 5, 6, 7, 8, 9, 10). Best score in bold

| minBERT/Task | SA | STS | PD |
|---|---|---|---|
| Base model | 0.399 | 0.003 | 0.376 |
| Full fine-tuned | 0.528 | 0.850 | 0.888 |
| Partial fine-tuned, merged | 0.525 | 0.862 | 0.885 |

Table 2: dev accuracy of minBERT models

| minBERT/Task | STS |
|---|---|
| Full fine-tuned, CS/MSE | 0.641 |
| Full fine-tuned, CE | 0.850 |

Table 3: STS loss objective comparison

| minBERT/Task | SST | STS | Paraphrase | Overall |
|---|---|---|---|---|
| Partial fine-tuned, merged | 0.524 | 0.844 | 0.885 | 0.751 |

Table 4: test leader board accuracy

The minBERT fine-tuned model performs poorly on the task of SA. No particular pattern is observed in the incorrect prediction results. The problem seems to be that of over-fitting. This is because the observed accuracy on the training data set is significantly higher than the one against the dev data set. Incorporating regularization techniques, increasing the drop out rate or increasing the size of the training corpora are some ways to alleviate the problem and improve accuracy.

The minBERT fine-tuned model works well on the task of PD. Upon inspecting the sentence pairs that were incorrectly predicted, it seemed as if the model ignored meaningful parts of the sentence when making predictions. Ex: *what are the best and profitable ways for saving money ?*, *what are your best ways to save money ?*. This behavior could be due to the use of the [CLS] token embedding from the last minBERT layer for training and prediction. Instead, a different pooling strategy, like mean pooling, could be evaluated for a more robust representation of sentences that would also improve performance.

## 7 Conclusion

I experimented with different extensions to the pre-trained BERT model that was developed in the first part of the project. I implemented the framework suggested in Wei et al. (2022) to generate model embeddings such that the top layers of BERT are fine-tuned specific to the task and the bottom layers are kept frozen to be shared across tasks. The partially fine-tuned merged model

provides the flexibility to update the model for frequently changing tasks without affecting other tasks, and still achieve similar performance as the task-specific fully fine-tuned models.

The results from my experiments were limited in that I was unable to thoroughly explore and optimize for the hyper-parameter space, i.e trying different drop out rates, batch sizes, etc..In addition, several other techniques like regularized optimization, Contrastive and Multiple Negatives Ranking Loss learning, and in-domain further pre-training can be conducted independently or in combination to boost performance.

I have learnt a lot as part of this project, from reading about the BERT architecture to building its core components and experimenting with extensions that have real world applications. Performing the entire cycle of model building, fine-tuning and evaluation has been an extremely rewarding experience.

## References


Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *SEM 2013 shared task: Semantic textual similarity. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 32–43, Atlanta, Georgia, USA. Association for Computational Linguistics.

Cristian Buciluundefined, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, page 535541, New York, NY, USA. Association for Computing Machinery.

Zihang Chen, Hongbo Zhang, Xiaoji Zhang, and Leqi Zhao. 2017. Quora question pairs.

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*, pages 2790–2799.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. 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 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

Amil Merchant, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney. 2020. What happens to BERT embeddings during fine-tuning? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 33–44, Online. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *CoRR*, abs/1908.10084.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Sam Bowman, Dipanjan Das, and Ellie Pavlick. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. In *International Conference on Learning Representations*.

Alex Wang, Jan Hula, Patrick Xia, Raghavendra Pappagari, R. Thomas McCoy, Roma Patel, Najoung Kim, Ian Tenney, Yinghui Huang, Katherin Yu, Shuning Jin, Berlin Chen, Benjamin Van Durme, Edouard Grave, Ellie Pavlick, and Samuel R. Bowman. 2019. Can you tell me how to get past sesame street? sentence-level pretraining beyond language modeling. In *Proceedings*

*of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4465–4476, Florence, Italy. Association for Computational Linguistics.

Tianwen Wei, Jianwei Qi, and Shenghuan He. 2022. A flexible multi-task model for BERT serving. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 785–796, Dublin, Ireland. Association for Computational Linguistics.