Adapting BERT for Multi-Task Learning with PALs

Stanford CS224N Default Project

Kathleen Cheng
Department of Computer Science
Stanford University
kcheng18@stanford.edu

Abstract

We aim to adapt BERT to perform multiple sentence-level tasks (sentiment analysis, paraphrase detection, and semantic textual similarity) simultaneously while sharing most model parameters between tasks. To achieve this, we added task-specific layers in parallel with the BERT layers in the original model. This adaptation did not improve performance, possibly due to a lack of training data. However, the "annealed sampling" approach we took to schedule tasks during training did improve the scores. This method accounted for the difference in sizes between datasets and changed the proportion of training dedicated to each task over time so that the model would neither overfit smaller datasets nor underperform on the associated tasks by not training on them enough.

1 Key Information

- 224N Staff Mentor: Hans Hanley
- No external collaborators
- Not a shared project

2 Introduction

We want to adapt a BERT model to perform well on multiple natural language processing tasks simultaneously. The motivation behind creating a model that can multitask is to create more generalizable language representations. Hopefully, adapting BERT in this way will make its language embeddings less dependent on any specific task. The most simple way to achieve high performance on multiple tasks is to have separate weights for the entire model for each task, but this methodology requires storage of a large number of parameters, and additionally, does not contribute to creating a model that can generalize well. Instead, we want adaptations to the BERT model that preserve a large number of shared parameters. In our approach, we share all weights in the original BERT model across tasks, and we incorporate additional task-specific projected attention layers (PALs) (Stickland and Murray, 2019) in parallel with the original BERT layers.

We also test an "annealed sampling" approach to sampling tasks during training. The datasets corresponding to our tasks of interest are very different in size, and this algorithm attempts to solve the issues that other methods have of either overfitting to smaller datasets or conversely, under-performing on the associated task due to a lack of training.

3 Related Work

Stickland and Murray (2019) attempted to design an adaptation to a BERT model that preserved a large number of shared parameters. They added separate task-specific layers in parallel with the original BERT layers, and they also proposed a new multitask training schedule for dealing with unevenly sized datasets. The idea of adding a small number of parameters for each task has also adopted by others in different ways. For example, Houlsby et al. (2019) added extra task-specfic layers in between the original BERT layers as opposed to in parallel.

We attempt to replicate Stickland and Murray (2019)'s results; after training on 8 different GLUE tasks (Wang et al., 2018), they achieved 92.6% accuracy on the sentiment analysis task, 71.5% accuracy on paraphrase detection, and 85.5% correlation on semantic textual similarity.

4 Approach

In order to adapt the BERT model to perform multiple tasks, we added task-specific projected attention layers (PALs) in parallel to the BERT layers as outlined in Stickland and Murray (2019). We also implemented the "annealed sampling" training schedule they introduced in order to incorporate training data from all datasets.

4.1 Model Architecture

4.1.1 Baseline BERT Model

The architecture of the original BERT model is described in Vaswani et al. (2017), and the methods used to pretrain its weights are detailed in Devlin et al. (2018). To adapt BERT to perform multiple tasks, we add a separate head specific to each task on top of the BERT model. This head takes as input the pooled output of BERT corresponding to the [CLS] token and applies a linear layer that outputs one logit for each possible label. This model was trained on just one task, sentiment classification, to verify the baseline implentation. Then, it was trained on all three tasks in order to serve as a baseline to further adaptations.

4.1.2 Projected Attention Layers (PALs)

As our adaptation to the model, we add a task-specific layer in parallel with each of the original 12 BERT layers.

If we define the self-attention (SA) function as

$$\text{SA}(\mathbf{h}) = \text{FFN}(\text{LN}(\mathbf{h} + \text{MH}(\mathbf{h}))) \tag{1}$$

where FFN is a feed-forward network, LN is a layer-norm, and MH is multi-head attention, each of the BERT layers in the original model can then be described as

$$\mathbf{h}^{l+1} = \text{LN}(\mathbf{h}^l + \text{SA}(\mathbf{h}^l)) \tag{2}$$

where l refers to the layer number. In words, each BERT layer applies an self-attention layer and then a layer-norm with a residual connection. In order to incorporate the task-specific layers, Equation 2 can be updated to

$$\mathbf{h}^{l+1} = \text{LN}(\mathbf{h}^l + \text{SA}(\mathbf{h}^l) + \text{TS}(\mathbf{h}^l)) \tag{3}$$

Following the projected attention layer (PAL) approach that Stickland and Murray (2019) introduced, the task specific layers take on the form

$$\text{TS}(\mathbf{h}) = V^D \text{MH}(V^E \mathbf{h}) \tag{4}$$

where V^E is a $d_s \times d_m$ "encoder" matrix, and V^D is a $d_m \times d_s$ "decoder" matrix. $d_m < d_s$, and each task has the same V^D and V^E across all layers in order to limit the number of task-specific parameters. Specifically, $d_s = 768$ is the size of the hidden layers in the original BERT model, and we choose $d_m = 204$ as the hidden layer size for multi-head attention within the PAL. Figure 1 illustrates how the task-specific PALs are incorporated into the model.

4.2 Training Schedule

Each training iteration uses a batch of samples from one of the datasets. The way we choose the next dataset to sample from can influence the multitasking ability of the model.

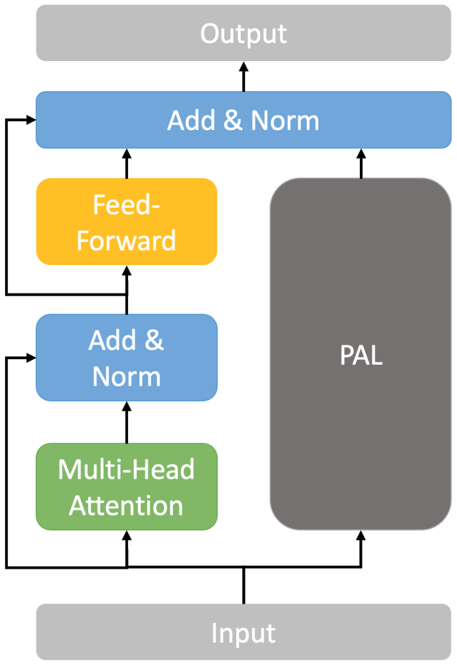

Figure 1: Schematic of a single BERT layer with a task-specific projected attention layer (PAL). The model consists of 12 sequential BERT layers.

4.2.1 Round Robin

As a baseline, a round robin approach can be used to incorporate datasets for multiple tasks while training a single model. Using this method, we cycle through each dataset in order, and therefore each dataset is sampled from an equal amount of times no matter its size.

4.2.2 Annealed Sampling

When using a round robin approach, the model can overfit on the smaller datasets, because by the time all of the samples in a larger dataset have been used, the examples in the smaller dataset will have been seen many times already. To remedy this issue, the size of each dataset should be considered when deciding which task to train on next. Specifically,

$$p_i \propto N_i^{\alpha} \tag{5}$$

where p_i is the probability of selecting a batch of examples from task i and N_i is the number of training examples for task i. We define

$$\alpha = 1 - 0.8\frac{e-1}{E-1} \tag{6}$$

where e is the current epoch and E is the total number of epochs. Towards the beginning of training, tasks with more examples are selected more frequently in order to avoid repeating examples for tasks with less examples and therefore overfitting. As the number of training epochs increases, the dependence on number of examples decreases and the multiple tasks are selected more equally. The motivation behind this "annealed sampling" method is that training on the same task for many steps can lead to decreased performance on other tasks, and this consideration becomes most important towards the end of training.

5 Experiments

5.1 Data

Each dataset is listed along with the combined size of the training and validation sets, the associated task, and the format of the labels.

- Stanford Sentiment Treebank (SST) Dataset (9,645 examples): sentiment analysis; labels are negative, somewhat negative, neutral, somewhat positive, or positive
- CFIMDB Dataset (1,952 examples): sentiment analysis, binary labels (negative or positive)
- Quora Dataset (161,710 examples): paraphrase detection, binary labels (1 for paraphrase, 0 for not a paraphrase)
- SemEval STS Benchmark Dataset (6,903 examples): semantic textual similarity, labels are 0 (not related) through 5 (same meaning).

5.2 Evaluation method

The two sentiment analysis tasks and the paraphrase detection task are evaluated based on accuracy given the categorical nature of the output. We use Pearson correlation for the semantic textual similarity task, since it is treated as a regression problem.

5.3 Experimental details

To verify the base BERT model on a single task (namely, sentiment analysis), it was trained and evaluated on the SST and CFIMDB datasets separately. In "pretrain" mode where the base BERT pretrained weights were frozen, a learning rate of 1×10^3 was used, and in "finetune" mode where all weights were finetuned, a learning rate of 1×10^5 was used. The batch size was 64 for the SST dataset and 8 for the CFIMDB dataset, and 10 epochs were run. The dropout probability was 0.3.

To test the multitasking functionality of BERT, we had four models with all combinations of the two model architectures and two training schedules discussed earlier. We trained these models on the SST, Quora, and STS datasets, and all weights were finetuned. Cross-entropy was used as the loss function for SST given that there were 5 categorical outputs, and similarly, binary cross-entropy was used as the loss function for Quora given the 2 categorical outputs. Since STS was treated as a regression problem, MSE loss was used.

The learning rate was 1×10^5, and the batch size was 32. 25 epochs with 1200 training steps each were run, and the dropout probability was 0.3. If took roughly 3 hours to train each model on an NVIDIA A10G Tensor Core GPU using these hyperparameters.

Additionally, we also tried to augment the SST dataset with the examples from the CFIMDB dataset. In order to maintain the batch size at 32, the CFIMDB examples had to be truncated. We trained the BERT model with PALs using annealed sampling on this augmented dataset to see if it would improve the sentiment analysis score.

5.4 Results

Table 1 shows the results for the original BERT model when trained on the SST and CFIMDB datasets. All accuracies were sufficiently close to the baseline accuracies given.

Dataset	Pretrain or Finetune	Baseline Accuracy	Model Accuracy
SST	Pretrain	0.390	0.396
CFIMDB	Pretrain	0.780	0.763
SST	Finetune	0.515	0.527
CFIMDB	Finetune	0.966	0.959

Table 1: Sentiment analysis accuracy results for the base BERT implementation compared to baselines provided in the handout.

Figure 2 shows the results for all combinations of the two model architectures and two training schedules implemented for multitasking. The best performing model did not use PALs and selected training samples using an annealed sampling approach. This model was trained on the previous training and validation sets combined, and it yielded a test accuracy of 0.5308 for SST and 0.8859 for Quora. The STS correlation was 0.8719, leading to an average score of 0.7629.

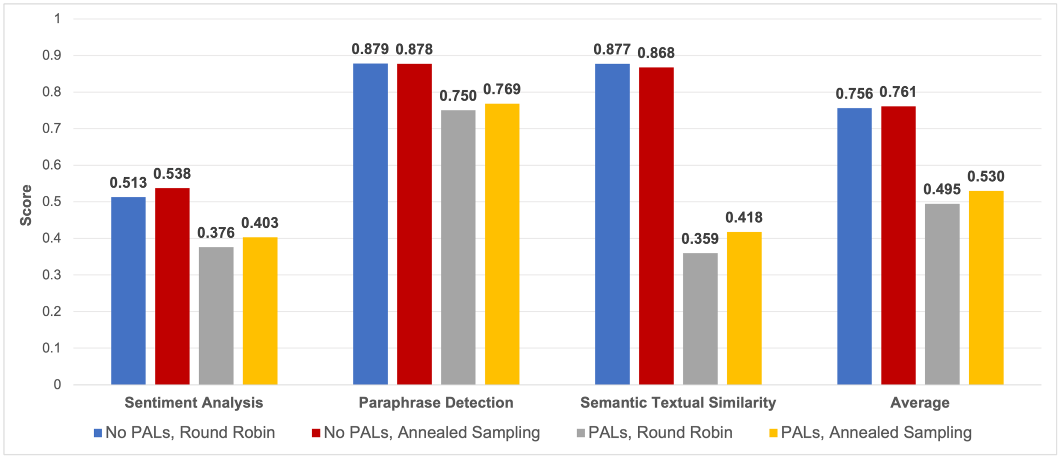

Figure 2: Comparison of models with and without PALs, and trained using either a round robin or annealed sampling approach. Accuracy was used to score the sentiment analysis and paraphrase detection tasks, and correlation was used for semantic textual similarity.

Because the sentiment analysis task performed worse than the other two tasks, we also tried to include the CFIMDB examples in training as described earlier. The results are compared to models trained using the original SST dataset in Figure 3. Adding the CFIMDB examples did not increase performance on the sentiment analysis task.

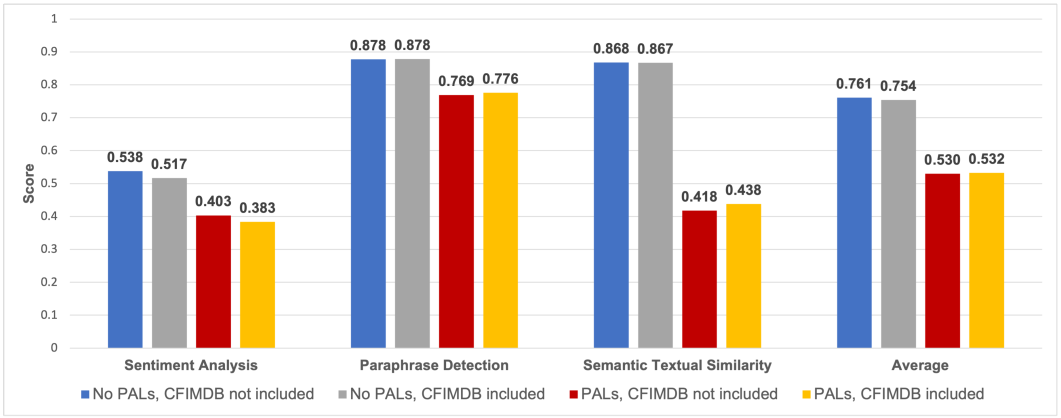

Figure 3: Comparison of performance on sentiment analysis, paraphrase detection, and semantic textual similarity when the original SST dataset is used for training vs. when examples from the CFIMDB dataset are added.

6 Analysis

Contrary to what Stickland and Murray (2019) reported, adapting the BERT model with task-specific PALs did not increase performance on any of the tasks. One reason might be that we did not train on the same volume of data that they did; instead of 8 tasks, we only used data from 3 tasks, and the Quora examples were a subset of the full dataset. Unlike the base BERT model which was pretrained on a very large corpus (3,300M words) Devlin et al. (2018), the PALs did not have pretrained weights, so a lot of data and training iterations would be required for them to improve the model's performance.

Assuming that we did not have enough data, it makes sense that adding in extra layers would only confound the model's predictions.

Interestingly, the sentiment classification task corresponding to the SST dataset had much lower scores than the two other tasks. This could be due to the fact that the paraphrase detection and semantic textual similarity tasks were very similar; in both cases, the model has to gauge how similar two sentences are in meaning. The size of these two datasets far outweighed the size of the SST dataset, so it is possible that the model fixated on these two tasks. Adding in the CFIMDB dataset actually worsened the performance on the SST dataset, and this could be because of the discrepancies in labels between the two datasets: SST labels are more fine-grained (negative, somewhat negative, neutral, somewhat positive, or positive), so adding in the CFIMDB dataset, which only has binary labels (negative or positive) may have caused these extreme labels to be predicted more often than they should have been.

7 Conclusion

In conclusion, we adapted BERT to perform multiple tasks by adding in task-specific PALs and adopting an annealed sampling training schedule. The PAL adaptation was compared to the original BERT model with multitasking heads, and the annealed sampling approach was compared to round robin training. Annealed sampling led to higher performance for models with and without PALs, and it seems to be a good approach to multitask training when the datasets are uneven in size. PALs led to lower performance no matter the training schedule, possibly due to a lack of training data. Without PALs, the baseline model architecture was able to perform the three tasks well already, which points to the generalizability of the sentence embeddings it produced with shared weights.

In the future, we could try to improve the performance of the BERT model with PALs by using more training data and more training iterations. Different architectures for task-specific layers and different ways of incorporating them into the original model could also be considered.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

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. *CoRR*, abs/1902.00751.

Asa Cooper Stickland and Iain Murray. 2019. BERT and PALs: Projected attention layers for efficient adaptation in multi-task learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5986–5995. PMLR.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *CoRR*, abs/1804.07461.