Implementation of BERT with Projected Attention Layers and Its Effectiveness

Stanford CS224N Default Project

Dayoung Kim
Department of Computer Science
Stanford University
dkim9613@stanford.edu

Wanbin Song
Department of Computer Science
Stanford University
wanbins@stanford.edu

Abstract

Multi-task learning is the one of interesting research field in natural language processing, where aiming for learning different tasks at the same time. There are several researches of multi-task learning area, targeting for maintain results as good as state-of-art methods which are trained on single tasks, and thus reduce the number of parameters used by the models. In this project, we implemented a multi-task BERT(Devlin et al., 2019) model for three NLP tasks, which are Sentiment Analysis, Paraphrase Detection, and Semantic Textual Similarity, trained and evaluated with the Stanford Sentiment Treebank (SST), the Quora Dataset and the SemEval Benchmark Dataset, respectively. To complete our implementation, we conducted two separate phases. Firstly, we created the baseline model, which is essentially a BERT model with a transformation added for every task based on the sentence representations that BERT provides. Additionally, we test the system by constructing three Projected Attention Layers (PALs) by (Stickland and Murray, 2019), one for every task, and distributing weights among the layers. By using different training methods and a model architecture from task-specific models, this approach aims to improve transfer learning while requiring fewer additional parameters.

1 Key Information to include

- Mentor: None
- External mentor: None
- Sharing project: None

2 Introduction

The advent of pre-trained models like BERT (Bidirectional Encoder Representations from Transformers) has impacted the discipline of natural language processing (NLP) dramatically. These models learn rich language representations through unsupervised pre-training on large text corpora. These representations can then be optimized for various natural language understanding tasks, leading to state-of-the-art outcomes. Unfortunately, this method usually requires fine-tuning a different model for every task, which results in an abundance of resource- and computational-intensive models. This challenge is particularly problematic when the application requires quick processing of several tasks at once or when there are limited computational resources available, like in the case of mobile devices. One potential solution for reducing these challenges is the idea of multi-task learning. Multi-task learning employs the domain-specific information obtained from the training data of related tasks to enhance the generalization performance of models. It is not easy to integrate multi-task learning with large pre-trained models such as BERT, despite its potential. Adapting a single model to multiple tasks efficiently while preserving performance is the primary challenge. This necessitates finding an

ideal balance between shared and task-specific parameters.
Various strategies for multi-task learning with BERT have been investigated in the last few years in this field. Projected Attention Layers (PALs) are a novel technique for efficient adaptation in multi-task learning that (Stickland and Murray, 2019) introduced. They work by adding low-dimensional multi-head attention layers in parallel to the BERT layers that are already in place. With a small number of task-specific parameters, this method allows a large portion of the model's parameters to be shared across tasks, making the model more parameter-efficient without significantly lowering task performance. By sharing parameters across tasks, we intend to increase accuracy on each one while lowering the overall number of parameters trained, which will lower the amount of memory needed for the model. These optimization objectives are essential for usage in target devices with limited memory, where typical machine learning model is required.

3 Related Work

In this section, we describe the related work of multi-task learning and typical challenges of implementing it.

3.1 Multi-task Learning

Multi-task learning is a famous paradigm in machine learning community, particularly within the NLP domain, aiming to improve learning efficiency and prediction accuracy by leveraging the commonalities and differences across tasks. Caruana (Caruana, 1993) provided an early exploration into multi-task learning, highlighting its potential for improving model generalization by sharing representations between related tasks. In the context of deep learning, multi-task learning has been facilitated by the architectural flexibility of neural networks, allowing for shared layers that capture generic representations and task-specific layers that learn task-dependent features.
The introduction of BERT and similar transformer-based models has shifted the focus towards leveraging pre-trained language representations for a wide array of NLP tasks. The standard approach involves fine-tuning the entire model on task-specific datasets, which, while effective, results in a separate model for each task. This approach is resource-intensive and not ideal for applications requiring the simultaneous handling of multiple tasks. Houlsby et al. (Houlsby et al., 2019) proposed a method for efficiently fine-tuning BERT for multiple tasks by adding adapter modules within the transformer layers, enabling task-specific fine-tuning with minimal additional parameters.
Stickland Murray (Stickland and Murray, 2019) furthered this line of research by introducing Projected Attention Layers (PALs), which add low-dimensional attention mechanisms in parallel to existing BERT layers for multi-task adaptation. This method allows for efficient parameter sharing across tasks while retaining the capacity for task-specific adaptations, aligning with efforts to make BERT more adaptable and efficient for multi-task scenarios.

3.2 Scheduling

To train multiple tasks simultaneously, we need to consider how we should iterate the task as well as associated dataset. Round-robin sampling, which involves going through the many activities one after another, is the conventional approach for multitask learning. The challenge arises from the fact that if a task appears in the dataset more frequently than another, it will repeat all of the task examples with few data points several times before repeating all of the other task examples. For the task that repeats frequently, this will result in overfitting, and for the other task, underfitting. In considering this, Stickland Murray (Stickland and Murray, 2019) suggest an innovative approach for training scheduling. The tasks are initially sampled in accordance with the size of their training set, but later, in order to minimize interferences, the weighting is lowered and the tasks are sampled more consistently.

4 Approach

The minBERT implementation is completed based on the default project code provided.

4.1 Base-line Multitask BERT

The multi-task BERT model is implemented based on the minBERT model. All tasks share same feed forward function. When forwarding input with attention mask, the model uses the base BERT layer with a dropout layer. The attention mask distributes the tokens whether they need to be ignored or not. The dropout layer is used to avoid overfitting by excluding some nodes. Using this forward function, each training functions are explained below.

- **Sentiment classification**: In this problem, the input is feed forwarded with the function mentioned above, then fed to an output layer. There are 5 target sentiment classes which are 0 as "negative", 1 as "somewhat negative", 2 as "neutral", 3 as "somewhat positive", and 4 as "positive". Therefore, the output layer is set to produce 5 logits for each class. For loss function, cross entropy is applied.
- **Paraphrase detection**: In the training process of paraphrase detection, each sentence in an input pair is forwarded separately, then combined to be fed into the output layer. The output layer is set to produce 1 logit and it passes it to the sigmoid function. The sigmoid function finally gives the probability of two sentences being paraphrases. The loss function that's used is binary cross entropy.
- **Semantic textual similarity**: For semantic textual similarity, an input pair of sentences are feed forwarded separately and two outputs are combined for the output layer like the paraphrase detection problem. Then, the output layer also outputs 1 logit implying how similar the sentences are. Finally, the mean squared error (MSE) is used to calculate the loss.

4.2 Scheduling

We implemented adaptive task sampling strategy for multi-task learning, dynamically adjusting task selection probabilities based on dataset sizes and training progress, as described in Section 3.2. Initialization calculates dataset sizes and establishes iterators for each task's dataset. The sampling probability adjustment employs an alpha parameter, which change with each epoch as below, ensuring a transition from size-based to uniform sampling.

$$\alpha = 1 - 0.8 \times \frac{epoch - 1}{epochs - 1}$$

Probabilities are calculated as the normalized dataset sizes raised to the power of α, facilitating a balanced task selection. This approach mitigates training bias towards larger datasets, promoting equitable learning and potentially enhancing model generalization across tasks.

4.3 PAL(Projected Attention Layers)

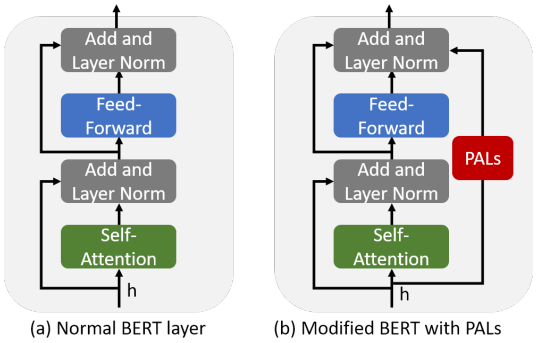

Figure 1: Architecture comparison BERT and PALs

As described in Related work, we are applying projected attention layers (PALs) to make use of multi-task learning to update BERT rather than fine-tuning individual tasks. As a first step, we set up BERT layer norm class based on the description in Stickland and Murray (2019). The layer norm

is ultimately used in PAL layer. Another layer we use is task specific function, which is PAL in the below expression. The expressions are as below with h as a hidden vector and H as the number of hidden units in a layer.

- **Standard feed-forward network**: $FFN(h) = W_2 f(W_1 h + b_1) + b_2$
- **Statistics used in layer normalization**: $\mu = \frac{\sum_{i=1}^{H} a_i}{H}$ $\sigma = \sqrt{\frac{\sum_{i=1}^{H}(a_i - \mu)^2}{H}}$
- **Concatenated attention heads**: $MH(h) = W \circ [Attention_1(h), ..., Attention_n(h)]$
- **Self-attention layer**: $SA(h) = FFN(LN(h + MH(h)))$
- **Projected attention layer**: $PAL(h) = V_D(SA(V_E(h)))$

In BERT with PAL layer, we only apply task specific function on the top. Each tasks share parameters in other layers, but in this one, the parameters are task-specific. It encodes the input with low rank size of 132, applies self attention function, then decodes the output to hidden size of 768. Sharing all but only having one layer different for each task could reduce the total number of parameters.

4.4 Optimizer

We used the Adam optimizer are described in the Default Project Handout. We implemented the step function of the Adam optimizer, all of which are also described in the Default Project Handout.

5 Experiments

5.1 Data

The data used for this project are Stanford Sentiment Treebank dataset (SST), movie reviews (CFIMDB) dataset, Quora dataset, and SemEval STS Benchmark dataset (STS). The SST dataset contains 11,885 sentence reviews having each review labelled as negative, somewhat negative, neutral, somewhat positive, or positive. The CFIMBD dataset consists of 2,434 movie reviews, each labelled as positive or negative. The Quora dataset has 400,000 question pairs with binary labels of whether each is paraphrase of one another or not. The STS dataset consists of 8,628 sentence pairs with label indicating the similarity on a scale from 0 to 5. We tokenized each dataset to make them suitable for our model. Sentences are split into word pieces and constant or unknown tokens are assigned. The output of this tokenization is the input of BERT embedding layer, the beginning of BERT training.

5.2 Evaluation method

We use accuracy for sentiment analysis and paraphrase detection. The accuracy is a metric for classification problems. It is the fraction of correct predictions among all predictions. On the other hand, for semantic textual similarity, we use Pearson correlation of the true similarity values against the predicted similarity values. Pearson correlations measures linear correlation between two sets of data.

5.3 Experimental details

For all of our experiments, we chose to train with a leaning rate of 10^{-5}, and a batch size of 16 for epoch 10. The hidden dimension was always 768 same dimension as our BERT embeddings. We also experimented with a larger batch size on frozen BERT with PAL, which was 32, but didn't give much difference. The accuracy compared to the batch size 16 was SST +0.001, paraphrase -0.006, STS +/-0.

5.4 Results

As you can see in Table 1, before applying PAL, for the dev score the sentiment classification training accuracy with SST dataset was 0.493, the paraphrase detection training accuracy with CFIMDB dataset was 0.741, and the semantic textual similarity training correlation score with STS dataset was 0.358, thus the overall training score was 0.638, without applying PAL. By freezing BERT parameters, which indicated in 'Pretrain' in the table 1, the overall score is slight lower but it took

approximately 4 times faster for training. By applying PAL to multi-task training, we could observe

Model	SST	Paraphrase	STS Corr	Overall Score
Base-line Frozen BERT	0.303	0.643	0.203	0.516
Base-line Fine-tuned BERT	0.493	0.741	0.358	0.638
Base-line+PAL Frozen BERT	0.308	0.643	0.215	0.520
Base-line+PAL Fine-tuned BERT	0.513	0.780	0.372	0.660

Table 1: Performance comparison of different tasks on SST, Paraphrase, and STS datasets.

the performance increment regardless of freezing BERT parameters. As you can see in Table 1, our best model shows the overall score as 0.660, which is 0.022 higher than without applying PAL.
For the test leaderboard, we got test accuracy of the sentiment classification training 0.532, the paraphrase detection as 0.785 and the semantic textual similarity correlation score as 0.075 which ends up with the overall test score with 0.618.

6 Analysis

As mentioned in Section 4.3, instead of using 3 different models, we only added on task specific layer on top of the model. Therefore, we need less parameters and reduce the total number of models used for multi-tasking. Also, because we share the parameters in other layers, we could get the better performance than BERT without PAL. The frozen BERT and fine-tuned BERT with PAL all showed better performances than the ones without PAL.

- **Sentiment classification**:

  Data 1: -LRB- Lawrence bounces -RRB- all over the stage, dancing, running, sweating, mopping his face and generally displaying the wacky talent that brought him fame in the first place .
  Label: 3 | Prediction: 1

  Data 2: A compelling Spanish film about the withering effects of jealousy in the life of a young monarch whose sexual passion for her husband becomes an obsession .
  Label: 3 | Prediction: 3

From sentiment classification results, there are some failures detected. For instance, Data 1 above, the model predicted it as "somewhat negative" but it was a "somewhat positive" comment. We assumed that the model might have viewed "sweating", "mopping", and "wacky" as negative words or it may not have seen "wacky" word before. The Data 2 is an example of successful predictions.

- **Paraphrase detection**:

  Data 1: What makes one angry? **&** What is the one thing that makes you angry?
  Label: 1 | Prediction: 0

  Data 2: Do you prefer to ask or answer questions on Quora? **&** Do Quora Users prefer to ask questions or to answer them?
  Label: 1 | Prediction: 0

  Data 3: How can I lock my pc browser (by pin) a specific webpage like youtube? **&** How can I specifically update a section of my webpage without PHP (I believe GitHub does not support PHP)?
  Label: 0 | Prediction: 0

The paraphrase detection also showed some failures in data like Data 1 and Data 2 above. For both of them, we decided that there's a chance that the model confuses with the grammer order easily.

- **Semantic textual similarity**:

  Data 1: "A cost analysis is under way", said Michael Rebell, CFE's executive director. **&** "We're not looking for a Robin Hood remedy", said Michael Rebell, the campaign's executive director.
  Label: 2.0 | Prediction: 0.007204

  Data 2: A man is mowing a lawn. **&** A woman is cutting a lemon.
  Label: 0.2 | Prediction: 0.150067

Among semantic textual similarity results, Data 1 shows what kind of sentence fails to be predicted will. When the sentence contains words of metaphore or abbreviation, there's a high chance that the model doesn't understand it. "Robin Hood remedy" would be the word that the model haven't seen before and don't understand. Therefore, the model would have difficulty of deciding similarity.

7 Conclusion

We implemented base-BERT model and conducted multi-tasking by using separate model for each task. Then, we applied PAL to use one model for all tasks. From the implementation of PAL, the parameters are shared for all tasks except for the task specific attention. This enables the effective usage of parameters. In addition, we had to deal with dataset scheduling. We applied annealed sampling with alpha parameter that changes from every epoch to train all datasets equally as possible. After fine-tuning our model, we could see the performance gets better. We successfully improved the multi-tasking performance, but there were some limitations of our work. One of our work's limitation is the variance we did not have enough tests on tweaking network architecture and tuning the hyperparameters. Future improvements include adopting various combinations of layers and do some data augmentation.

8 Contribution

- **Dayoung Kim**: Implementation of PAL and write the final report.
- **Wanbin Song**: Implementation of minBERT and write the final report.

References

R Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 41–48. Citeseer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

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*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

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.