Default Project: minBERT and Downstream Tasks

Mabel Jiang
Department of Mechanical Engineering
Stanford University
jyh1998@stanford.edu

Rachel Yu
Department of Biomedical Data Science
Stanford University
yuyue01@stanford.edu

Abstract

The goals of this project are to gain a deeper understanding of BERT's architecture, evaluate its performance on different datasets and tasks, and explore fine-tuning methods to enhance its performance. Specifically, we implement BERT architecture on sentiment classification task first, which serves as a backbone for the project baseline. Then we expand the model to multi-task classifier and examines its performance on its multi-task capability in sentiment classification, paraphrase detection, and semantic textual similarity tasks. To solve the conflicting gradients challenge and unbalanced dataset issue in multi-task classifier, we attempt various methods, like round robin and gradient surgery. We also seek different combinations of model techniques, like logits calculation and loss function. The finetuned model employing gradient surgery and cosine similarity outperforms the baseline model and other finetuned models as well. Overall, our finetuned model achieves a decent performance in the three tasks, especially in textual similarity task, and, with some future work, has the potential to become a robust and adaptable BERT-based multi-task model.

1 Introduction

From chatbots to virtual assistants and language translation tools, NLP has revolutionized the way we communicate with machines. Additionally, NLP techniques such as sentiment analysis, text classification, and information extraction have proven useful in areas such as healthcare and marketing [1]. More specifically, language model pre-training has been shown to be effective for improving many natural language processing tasks. One of the effective ways to apply pre-training to downstream tasks is Bidirectional Encoder Representations from Transformers (BERT) model, whose details can be found in Section 2.

This project aims to gain a deeper understanding of the BERT model architecture by implementing its key components. In other words, the project attempts to demonstrate the capability of the BERT model for performing NLP tasks and evaluate its performance on different datasets and downstream tasks. Section 3 talks about our baseline model, which is a BERT model loaded with pre-trained weights and sentence classification were performed on two datasets. Section 3 also examines how to fine-tune BERT's contextualized embeddings to simultaneously perform well on multiple sentence-level tasks including sentiment analysis, paraphrase detection, and semantic textual similarity. Various multi-task fine-tuning methods, like round robin and gradient surgery, were implemented. Our final fine-tuned model achieves an average score of 0.67, with similarity correlation being 0.773, and outperforms baseline in all three tasks. More details about experiments, results, and results analysis can be found in Section 4 and Section 5.

2 Related Work

With the development of advanced deep learning techniques, Natural Language Processing(NLP) has made significant progress in recent years. Several recent works have made significant contributions to NLP by leveraging the power of transformer-based models. "Attention Is All You Need" [2]

proposed the Transformer model, which introduced self-attention mechanisms to capture long-range dependencies in input sequences. It is based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. The paper showed that the Transformer outperformed previous state-of-the-art models on machine translation tasks while significantly reducing training time. Building upon the Transformer architecture, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" [3] introduced a pre-training approach that builds a deep bidirectional Transformer that can better capture the context of a word in both directions, instead of using pre-trained unidirectional or shallow Transformers as other previous models did. Specifically, it is a new pre-training technique called masked language modeling (MLM). MLM is achieved by masking a proportion of the tokens in the input, and then training the model to predict the masked tokens based on the context of the remaining tokens. This allows the model to learn bidirectional contextual representations of words, which can then be fine-tuned for specific NLP tasks. The BERT model outperforms previous state-of-the-art models on a variety of NLP benchmarks.

Although multi-task finetuning on language models is popular and many related researches have been done, the optimization challenges in such task is not fully understood yet [4]. A potential hypothesis that one of the key challenges in optimizing multi-task learning is the conflicting gradients from different tasks that impede progress [4]. In this context, gradients are deemed to be conflicting if they have a negative cosine similarity. The definition of conflicting gradients is shown in Figure 1.

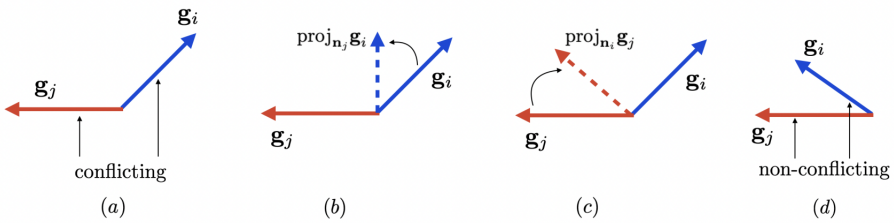

Figure 1: Projecting gradients. (a) shows conflicting gradients, (c) shows the projected gradients in conflicting case, (d) shows non-conflicting gradients) [4].

It's suggested that using a potential method, called Projecting Conflicting Gradients (PCGrad), a way of gradient surgery, can mitigate the effect of conflicting gradients [4]. PCGrad aims to modify gradients for each task to minimize negative conflict with other task gradients, while enabling positive interactions between them, without imposing assumptions on the model's form. Therefore, only conflicting gradients are altered while non-conflicting ones are left unchanged. More details about the implementaiton of PCGrad in BERT will be discussed in 3.3.

3 Approach

3.1 minBert

The backbone of the model that we use in this default project is BERT, which convert sentence input into tokens before it performs any additional processing. The BERT model first breaks down the input text into individual tokens and assigns each token a unique ID. Then, a trainable embedding layer is applied to each token. Besides embedding layer, BERT also includes transformer layer, which comprises multi-head attention, a residual connection with an additive and normalization layer, a feed-forward layer, and another residual connection.

$$MultiHead(Q,K,V) = Concat(head_1,...,head_h)W^O$$
$$\text{where}\, head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

Figure 2: Multi-head attention

Figure 2 shows the equation of multi-head attention. The output of BERT will be pooler classification/[CLS] token embedding. The starter code that we used is the one provided in the default project handout.

Table 1. minBERT results on SST and CFIMDB datasets

	Pretrain Dev Accuracy	Fine-tune Dev Accuracy
SST	0.403	0.511
CFIMDB	0.763	0.971

To establish the backbone of our baseline, we utilize the original BERT model, which incorporates multi-head self-attention and Transformer layers, and call it minBERT. We employ this minBERT model to conduct sentiment classification on the Stanford Sentiment Treebank dataset (SST) and the movie reviews dataset (CFIMDB).

The accuracy of both pretraining and fine-tuning the minBERT model on SST and CFIMDB dataset are shown in Table 1. They are very close to the provided accuracies in the default project handout, which are the mean reference accuracies over 10 random seeds with their standard deviation. Fine-tune accuracy is higher than pretrain accuracy, which makes sense because pretraining doesn't update the weights of BERT layers, while fine-tuning modifies BERT weights based on gradient, therefore fine-tuning the model will make it more suitable for this specific sentiment classification task. The accuracies of the model trained on SST is lower than those on CFIMDB. A possible reason is that the movie reviews in CFIMDB are highly polarized, which might be more easily classified than the not-that-polarized reviews in the SST dataset. Overall, because minBERT model achieve a decent performance, it is used as the backbone for our multi-task baseline model.

3.2 Baseline

With minBERT as our baseline backbone, we add linear additional layers on top of minBERT and customize method of calculating logits and loss function for each task respectively. For simplicity, sentiment classification task will be referred as "SST", paraphrase detection task as "para", and semantic textual similarity task as "STS". More specifically, in our baseline, we use cross entropy loss for sentiment classification, binary cross entropy loss for paraphrase detection, and MSE loss for textual similarity task. The details of additional layers and other model configurations added for each tasks can be found in Section 4.

3.3 Gradient Surgery

As mentioned in Section 2, when fine-tuning a pre-trained model on multiple tasks, we can observe that some of the gradients may be too large or too small, which can negatively affect the learning process. In our model, because of the sizes of the unbalanced datasets, the backpropagation of the para task, whose Quora dataset is the largest and has a size much larger than the other two datasets, will negatively impact the gradient of the other two tasks. Therefore, we implement PCGrad, a gradient surgery method, in our model to reduce the effect of conflicting gradients, whose definition can be in Section 2. We follow the procedure of PCGrad [4] as the original paper suggested: we first determine whether the current task's gradient are conflicting with other tasks' by calculating their cosine similarity; if the similarity is negative, the current task's gradient will be replaced by its projection onto the normal plane of its conflicting gradient; this procedure will be repeated for all three tasks respectively.

3.4 Cosine-similarity Fine-Tuning

In one of our fine-tuning approaches, we incorporate the cosine similarity into STS task training. It measures the similarity between two embeddings in a high-dimensional space. Cosine similarity ranges from -1 to 1, with values closer to 1 indicating a high degree of similarity. Specifically, during training steps, we finetune our multi-task model using cosine embedding loss, which is based on cosine similarity. It aims to learn the embeddings of words or phrases that are close in semantic meaning. The objective is to minimize the distance between the embeddings of similar items and maximize the distance between the embeddings of dissimilar items. The result is a scalar value that represents the loss, which is minimized during training.

4 Experiments

4.1 Data

For our baseline's backbone, minBERT model, we utilize two datasets, SST and the CFIMDB dataset, for our sentiment analysis experiments. The SST dataset contains 215,154 unique phrases extracted from parse trees, each of which has been annotated by three human judges. Meanwhile, the CFIMDB dataset comprises 2,434 highly polarized movie reviews, with each phrase assigned a label of negative, somewhat negative, neutral, somewhat positive, or positive.

To evaluate our model's performance on multi-tasks model, we make use of two additional datasets: Quora and the SemEval STS Benchmark Dataset. The Quora dataset consists of 400,000 question pairs, each labeled as either a paraphrase or not. The SemEval STS Benchmark Dataset contains 8,628 sentence pairs evaluated for similarity on a scale ranging from 0 (unrelated) to 5 (equivalent meaning). All datasets and their respective splits are provided as part of the default project. By leveraging these datasets, we were able to train and test our model's performance on multiple tasks.

4.2 Evaluation method

We use two evaluation metrics for the three tasks. The evaluation metric for SST and para tasks is accuracy, because it represents the proportion of correctly classified instances among all instances in the dataset and it is one of the simplest and most intuitive metric that indicates how well a model is able to classify the data. In terms of STS task, we use Pearson correlation coefficient to measure its performance (as handled by the default project starter code). It is a reasonable choice because it provides a single value that summarizes the overall linear correlation between the ground truth labels and the model's predicted similarity scores. We evaluate our fine-tuned multitask models performance using the above metrics and the scores that we compare them against are results of our baseline model.

4.3 Experimental details

In our experiments, we train the model for a total of 10 epochs and apply a learning rate of $1e-3$ for pre-training, followed by fine-tuning with a learning rate of $1e-5$. Additionally, we set the hidden_dropout_prob to 0.3 for all experiments. Prior to conducting the multi-task baseline experiment, we initially train three individual tasks in order to evaluate the efficacy of each dataset. The results, which are shown in section 4.4, indicate each task achieves each task attains a reasonably high accuracy during individual training.

To train our multi-task model, we utilize a round-robin method that involved zipping three data loaders and taking one batch from each loader in turn. To ensure that each dataset received comparable training, we set different batch sizes for each dataset according to its size. This approach requires re-weighting the losses per data sample for each task during training. In the baseline experiment, we use batch sizes of 8, 141, and 6 for SST, para, and STS, respectively. In subsequent fine-tuning models, we adjust the batch sizes to 2, 35, and 1, respectively, due to memory limitations on our GPU.

To optimize our model for each task, we customize configurations for each experiment, including techniques used, specific loss functions, and additional layers on top of our backbone. The detailed configurations for each experiment are shown in Table 2. Notably, we incorporate gradient surgery, as in Finetune 1 model in Table 2, to avoid interference between gradients from different tasks in our multi-task training. The motivation behind trying this gradient surgery method is that we observe that our baseline multi-task model performs worse than single-task experiments, particularly for the STS task, which has a much smaller dataset. To further improve the model's performance on the STS task, we use cosine-similarity fine-tuning in Finetune 2 model to improve our embeddings by utilizing CosineEmbeddingLoss to calculate the loss. In the last fine-tuning, referred as Finetune 3, we incorporate cosine similarity into the STS task by calculating STS logits as the cosine similarity score between two embedding outputs, rather than using CosineEmbeddingLoss directly.

Table 2. Model configurations for each experiment

Model	Techniques involved	Task	Additional Layer	Loss Function
Baseline	none	SST	linear(hidden_size, 5)	cross entropy loss
		para	linear(hidden_size*2, 1)	binary cross entropy loss with logits
		STS	linear(hidden_size*2, 1)	MSE loss
Finetune 1	gradient surgery	SST	same as Baseline	
		para		
		STS		
Finetune 2	gradient surgery + cosine-similarity fine-tuning	SST	same as Finetune 1	
		para		
		STS	linear(hidden_size, hidden_size)	cosine embedding loss
Finetune 3	gradient surgery + calculation of STS logits as cos similarity between embeddings outputs	SST	same as Finetune 1	
		para		
		STS	linear(hidden_size,100), linear(100,hidden_size)	MSE loss

4.4 Results

The dev results of four experiments evaluated per task are shown in Table 3. Our baseline model achieves much higher accuracy in para task than in SST and STS tasks. Our assumption is that because the Quora dataset for the para task is so much larger than the other two datasets, the gradients of the other two tasks are negatively impacted by the gradient of paraphrase. Therefore, to test this assumption, we run single-task experiments, whose results are 0.526 for SST, 0.698 for para, and 0.703 for STS. Given that results of SST and STS are notably better than that in our baseline multi-task model, our assumption is confirmed. Accordingly, we first try adding gradient surgery in Finetune 1 to reduce the negative impact.

Upon comparison of Finetune 1 to the baseline, it is evident that the incorporation of gradient surgery yields a substantial enhancement in both the SST and STS tasks. However, the para task exhibits a comparatively modest improvement. This outcome aligns with our anticipated result, as gradient surgery is intended to alleviate the adverse effects of conflicting gradients, resulting in greater performance improvements in tasks with a more pronounced negative impact on their gradients.

In our second endeavor, we implement the cosine embedding loss function to compute the loss for similarity task during training, utilizing three inputs: embedding 1, embedding 2, and labels. However, the outcomes are not in line with our expectations, as this approach adversely impacted the model's performance across all tasks. While the idea of integrating cosine similarity into the similarity task is reasonable, it appears that the loss function may have been improperly employed in our model, which warrants further exploration in future research. Consequently, we opt to adopt an alternate approach to incorporate cosine similarity into the model, which refers to the Finetune 3 model.

The Finetune 3 model exhibits outstanding performance across all three tasks, achieving scores of 0.515 for sentiment classification, 0.723 for paraphrase detection, and 0.773 for semantic textual similarity. Notably, the STS task demonstrates the most significant improvement in this experiment. This outcome is unsurprising, given that cosine similarity measures the alignment of two embeddings' directions in high-dimensional space, which is less susceptible to changes in document length and term frequency. Consequently, compared to utilizing logit from concatenated embeddings, utilizing a scaled cosine similarity logit as the prediction and incorporating an additional loss function to assess the logit during training is a more reasonable approach for similarity tasks. In addition, the test result of our best model, Finetune 3, is 0.521 for SST, 0.723 for para, and 0.751 for STS.

Table 3. Dev results for each experiment

Tasks	Metrics	**Baseline**	**Finetune 1** (+ gradient surgery)	**Finetune 2** (gs + cosine embedding loss)	**Finetune 3** (gs + cosine similarity)
SST	Accuracy	0.335	0.511	0.499	**0.515**
para	Accuracy	0.692	0.722	0.636	**0.723**
STS	Correlation	0.260	0.368	0.203	**0.773**

In summation, the collective findings suggest that the application of gradient surgery amplifies the efficacy of the multi-task model, and the utilization of cosine similarity serves to enhance the model's

performance on a designated similarity task.

5 Analysis

We conduct error analysis on each task by inspecting some outputs of our model. We observe that the model exhibited strong performance in accurately predicting sentiment for short sentences that contained clear positive or negative language. For example, the sentence "A quiet treasure – a film to be savored" is correctly predicted as positive (4). However, there are some short and neutral (2) sentences like "That is it" and "A delirious celebration of the female orgasm", which are predicted as positive (4). These reveal that the model tends to rely heavily on positive or negative language and lacks the ability to adequately interpret context or understand sarcasm, irony, idiomatic expressions, or figurative language. This point can also be indicated by other misclassifications, such as predicting a somewhat positive (3) sentence "It seems like I have been waiting my whole life for this movie and now I can't wait for the sequel.", as a negative (0). The "can't wait" is the idiomatic expression to express excitement and anticipation, where the model might lack enough samples to learn.

For paraphrase detection, we find that many false-positive cases are where the two sentences contain many same words in a similar sequence. For example, the two sentences are "Why do I fall asleep when sitting?" and "Why do I keep falling asleep?". The first sentence suggests that the person falls asleep while sitting, while the second sentence suggests that the person falls asleep repeatedly. Although the two sentences are similar with many common words, they are not exactly equivalent. But the model identifies them as paraphrases. This may be due to the fact that the model has not been trained on enough examples of similar but not identical sentences.

For semantic textual similarity, our findings indicate that the model performs well on short sentence pairs with simple grammatical structures, but struggles when presented with more complex language. Specifically, we find that the model is not effective at capturing polysemy in sentences. For example, the sample with sentences "Work into it slowly" and "It seems to work" has the ground truth similarity of 0.0, while the model predicts this sample with a similarity score of 2.36. This indicates an inability to differentiate between the different meanings of the word "work", so it may incorrectly identify the word "work" as a semantic similarity. In addition, we also observe that the model does not perform well on sentences containing negation. For instance, the similarity score between "Democracy is a threat to liberty" and "Democracy has nothing to do with liberty" is 1.8 while the model predicts them as having almost the same meaning with a similarity of 4.42. This indicates that the model is not able to differentiate between the negation in the second sentence. Overall, the model may struggle with identifying nuanced language and overemphasizes common concepts while disregarding subtle differences. Addressing this limitation may require the use of more diverse and nuanced training data to improve the model's ability to capture complex linguistic structures.

6 Conclusion

In conclusion, the integration of gradient surgery and cosine similarity into our multi-task model yields superior performance compared to the baseline and other fine-tuned models. This indicates the efficacy of gradient surgery in mitigating the conflict gradients arising from disparate tasks and the suitability of cosine similarity for similarity tasks. To enhance future experiments, it may be beneficial to investigate alternative finetuning techniques tailored for each specific task, such as incorporating supplementary pretraining, implementing multiple negative ranking loss, and utilizing regularized optimization. Additionally, augmenting the dataset to be learned by the model could be a valuable avenue to explore. Lastly, as current limitations exist in terms of GPU resources and memory, we could consider adjusting training duration and experimenting with other hyperparameters more frequently if greater GPU availability becomes feasible.

References

[1] "Sentiment Analysis: Concept, Analysis and Applications." by Gupta, Shashank. (2018)

[2] "Attention is All You Need" by Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser, and Polosukhin. (2017)

[3] "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" by Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova (Google AI Language, 2018)

[4] "Gradient Surgery for Multi-Task Learning" by Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, Chelsea Finn. (2020)