BEAKER: Exploring Enhancements of BERT through Learning Rate Schedules, Contrastive Learning, and CosineEmbeddingLoss

Stanford CS224N Default Project

Elizabeth Baena
Department of Computer Science
Stanford University
ebaena@stanford.edu
Mentor: Olivia Yee

Abstract

This paper has two primary research goals, which are 1) to determine the baseline performance of BERT with pre-trained weights on sentiment analysis using the SST and CFIMDB datasets and 2) to investigate the effects of a learning rate schedule, contrastive learning, and CosineEmbeddingLoss on the performance of BERT on sentiment analysis, paraphrase detection, and semantic textual similarity (STS). Examining methods to improve the performance of BERT will continue the research conversation on how to develop sentence embeddings that are robust and generalizable.

The approach utilized was to implement a baseline model of BERT called minBERT using multi-headed attention and the Adam Optimizer and then train this model for sentiment analysis. BEAKER is an extension to BERT that is trained on the sentiment analysis, paraphrase detection, and STS tasks using the SST, Quora and SemEval STS Benchmark datasets, respectively. From the experiments performed in this project, we discover that in particular, of the three methods explored, learning rate schedules with a warmup period have the potential to significantly improve the performance of BERT on these three salient NLP tasks.

1 Introduction

One of the core research goals in the field of Natural Language Processing (NLP) is to develop computational models that can accurately represent human language. This is a challenging topic for researchers as human language is complex and contains intricacies that can begin to be understood by models after a significant amount of training on text corpora. A language model that has gained significant attention from researchers in recent years is the BERT model, or Bidirectional Encoder Representations from Transformers model, proposed by Devlin et al. (2019) in 2019. The state-of-the-art performance of the BERT model demonstrates the potential of pre-training language models on unsupervised data to improve the quality of the sentence embeddings that are generated. In addition, a major contribution of Devlin et al.'s research is the discovery that extensive pre-training in combination with bidirectional model architectures leads to robust embeddings that can be used for a variety of downstream NLP applications. This project aims to study how BERT's performance can be enhanced for three salient NLP tasks: sentiment analysis, paraphrase detection, and semantic textual similarity (STS). Although the BERT model itself achieves high performance on various tasks, there is a great opportunity to research how to achieve better results on classification tasks by using techniques such as learning rate schedules, cosine similarity fine-tuning, and contrastive learning during training. The contributions of this paper are as follows:

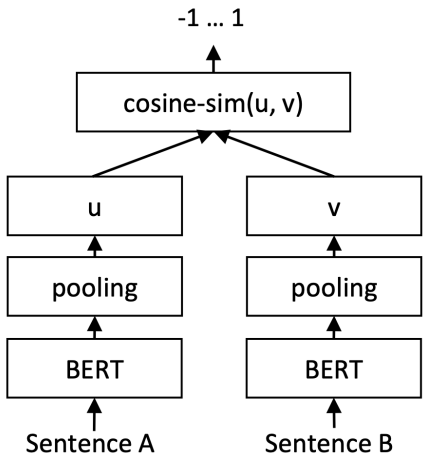

Figure 1: Architecture of SBERT using cosine-similarity (Reimers and Gurevych, 2019)

- The project proposes an enhanced BERT model which we name BEAKER that uses a linear learning rate schedule with a linear warmup to improve the performance of BERT on sentiment analysis, paraphrase detection, and STS.
- The project researches the effect of different temperatures for contrastive learning on the performance of the BERT multitask classifier.
- The BEAKER model with a linear learning rate schedule achieves greater performance on sentiment analysis, paraphrase detection, and STS tasks than the baseline multitask BERT model.

2 Related Work

2.1 Learning Rate Schedules

The paper Leveraging Pre-trained Checkpoints for Sequence Generation Tasks by Rothe et al. (2019) exemplifies the use of a learning rate schedule with warmup in a transformer language model. Rothe et al. developed a sequence-to-sequence model that achieves state-of-the-art results in sequence generation tasks such as machine translation (MT), text summarization, sentence splitting, and sentence fusion. Their model uses a linear warmup to the initial learning rate of 0.05 for the Adam Optimizer. This warmup schedule utilizes 40 thousands steps and has a square root decay down to a learning rate of 0. The success of using such a learning rate schedule for developing a language model for downstream tasks demonstrates the potential of learning rate schedules for other models like BERT. Rothe et al.'s publication was an inspiration for this project's extension to use a linear warmup and linear decay learning rate schedule to improve the performance of BERT on the classification tasks of sentiment analysis, paraphrase detection, and semantic textual similarity (STS).

2.2 Cosine Similarity Fine-Tuning

Another extension to the multitask BERT classifier was inspired by the paper Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks by Reimers and Gurevych (2019). In their publication, Reimers et al. proposes the model SBERT, which is a modification of BERT that uses cosine-similarity to compare the semantic meaning of two sentence embeddings using cosine-similarity. The architecture of SBERT is shown in Figure 1. SBERT applies BERT to two sentences and obtains the pooled output from each sentence to obtain u and v. Then, these pooled outputs u and v are compared using the cosine-similarity function, which returns a value from -1 to 1 representing the similarity score of the two sentences. The high performance of SBERT on STS tasks by using a cosine-similarity measure to predict the similarity of two sentence embeddings demonstrates the promise of using cosine similarity to fine-tune the multitask BERT model for STS.

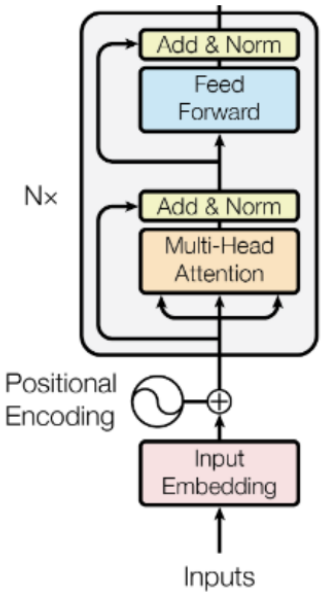

Figure 2: Transformer Encoder Layer of BERT model (Vaswani et al., 2017)

2.3 Contrastive Learning

As another extension, I drew inspiration from SimCSE: Simple Contrastive Learning of Sentence Embeddings by Gao et al. (2021). The authors had success on the semantic textual similarity (STS) task using a supervised approach with "entailment" pairs as positives and "contradiction" pairs as hard negatives. In contrastive learning, the intuition is to bring together semantically close neighbors and push apart non-neighbors.

3 Approach

3.1 Architecture of Baseline BERT model

The baseline BERT model follows the approach detailed in the original BERT paper and consists of embedding layers with a dimensionality of 768 (Devlin et al., 2019). The hidden state of the first token, or the [CLS] token, will represent the embedding of the whole sentence.

The BERT baseline model used in this project is implemented using 12 Transformer layers. Each Transformer layer applies multi-head attention followed by an additive and normalization layer with a residual connection. The transformer layer then applies a feed-forward layer and another additive and normalization layer with a residual connection. This can be visualized in Figure 2, which is taken from the paper Attention is All You Need (Vaswani et al., 2017).

For this project, I implemented the multi-head self-attention and Transformer layer according to the implementation details in Attention is All You Need (Vaswani et al., 2017). The equation for the scaled dot-product attention is $Attention(Q, K, V) = softmax\left(\frac{QK^\top}{\sqrt{d_k}}\right)V$. These attentions are concatenated along the heads to obtain the multi-head attention. We define the feed-forward network as $FFN(x) = \max(0, xW_1 + b_1)W_2 + b_2$. This applies a ReLU activation function followed by a linear layer. An important implementation detail is that a dropout layer with a value of $p_{drop} = 0.1$ is applied to each sub-layer's output before being added to the sub-layer input and before normalization. Dropout is also used on the sum of the embeddings with the positional encodings. Another part of the project that I implemented was the BERT classifier class, which uses the BERT model to encode sentences and then applies dropout on the pooled output of each sentence and projects it through a

linear layer in order to classify each sentence. Finally, I implemented the step() function of the Adam Optimizer from scratch using the efficient method of bias correction, which is performed through the following two lines of pseudocode:

$\alpha_t \leftarrow \alpha \cdot \sqrt{1-\beta_2^t}/1-\beta_1^t$

$\theta_t \leftarrow \theta_{t-1} - \alpha_t \cdot m_t/(\sqrt{v_t}+\epsilon)$

3.2 Architecture of the Enhanced BERT model, BEAKER

BEAKER uses exactly the same model architecture as BERT, including 12 Transformer layers, a hidden size of 768, multi-headed self-attention, and the Adam Optimizer with efficient bias correction.

All of the changes to BEAKER are in the training methods for the sentiment analysis, paraphrase detection, and semantic textual similarity (STS) tasks including adaptive learning rates, the use of CosineEmbeddingLoss for finetuning of BERT for the STS task, and contrastive learning with varied temperature parameters.

3.2.1 Learning Rate Schedule Implementation

In order to incorporate adaptive learning rates, this experiment uses the `get_linear_schedule_with_warmup` method from HuggingFace's `transformers` library. This establishes a learning rate schedule that has a linear decay from the initial learning rate that is passed in as an argument to the optimizer to a final value of 0. Before this learning rate decay, there is a warmup period where the learning rate increases in a linear fashion from a starting point of 0 to the learning rate that is passed in to the optimizer. Thus, an important part of this approach was to experiment with various initial values of the learning rate hyperparameter for the optimizer.

3.2.2 Cosine Similarity Fine-tuning for STS

This research also experimented with cosine similarity fine-tuning for the STS task. To implement this, the experiment uses PyTorch's CosineEmbeddingLoss when fine-tuning and cross-entropy loss for pretraining as is used for the pretraining and finetuning of the other two tasks. In order to use CosineEmbeddingLoss, the labels of the data must be normalized to a value between 0 to 1, with 0 signifying no similarity between the sentences and 1 signifying an equivalent meaning.

3.2.3 Contrastive Learning Implementation

This project's implementation of contrastive learning leverages the NT-Xent loss function proposed by Chen et al. (2020) in A Simple Framework for Contrastive Learning of Visual Representations:

$$\ell_i = -\log \frac{e^{\text{sim}(\mathbf{h}_i,\mathbf{h}_i^+)/\tau}}{\sum_{j=1}^{N} e^{\text{sim}(\mathbf{h}_i,\mathbf{h}_j^+)/\tau}},$$

Figure 3: NT-Xent loss function (Chen et al., 2020)

where i corresponds to the i-th batch, N is the minibatch size, h_i and h_i^+ are the embeddings of the training examples, τ denotes a temperature parameter, and sim(h_1, h_2) is the cosine similarity.

In my project's implementation, we obtain the pooler output from BERT. We pass the pooled output through a Linear layer with ReLU activation and a dropout layer with dropout probability $p_{drop} = 0.3$. The cross entropy loss between the cosine similarity of the two embeddings is computed.

3.3 Baselines

The scores of the baseline BERT model on the task of sentiment analysis are compared to the accuracy benchmark listed on page 14 of the CS 224N Default Final Project handout. The performance of the enhanced BERT multitask classifier with the extensions of learning rate schedules, cosine similarity

fine-tuning, and contrastive learning will be compared to the performance of the BERT multitask classifier I implement without any extensions.

4 Experiments

4.1 Data

The study both pretrains and finetunes the baseline Bert implementation on the Stanford Sentiment Treebank (SST) dataset, composed of 11,855 sentences from movie reviews (Socher et al., 2013). Additionally, it pretrains and finetunes on the CFIMDB dataset, consisting of 2,434 highly polarized film reviews (Choi et al., 2022). These datasets will be used for the NLP task of sentiment analysis, which classifies text into categories according to its polarity.

The experiment uses the Quora dataset for the paraphrase detection NLP task, which is the task of determining whether two pieces of text are paraphrases of each other. The Quora dataset contains 400,000 question pairs that are labeled according to whether one question is a paraphrase of the other (Sharma et al., 2019).

The SemEval STS Benchmark dataset is utilized for the NLP task of semantic textual similarity, or STS (Agirre et al., 2013). STS ranks the degree of similarity between two texts from a scale of 0, meaning no similarity, to 5, representing the same meaning. The SemEval dataset used for STS contains 8,628 sentence pairs.

4.2 Evaluation method

The study utilizes accuracy as the evaluation metric for the sentiment analysis task using the SST and CFIMDB datasets.

Accuracy is also the evaluation metric that will be used for the paraphrase detection task using the Quora dataset. Accuracy is chosen as a metric due to the binary labels of the task.

The study will use Pearson correlation as the evaluation metric for the STS task using the SemEval STS Benchmark dataset.

4.3 Experimental details

4.3.1 BERT Baseline Experiment

The baseline BERT model was trained on a NVIDIA T4 GPU using a learning rate of $1e^{-}3$ for pretraining and $1e^{-}5$ for finetuning. Training time takes approximately 15 minutes each for the SST and CFIMDB datasets. The baseline BERT model was pretrained and finetuned for 10 epochs each. These same learning rates were initially used for the initial BERT multitask classifier in order to maintain the learning rate hyperparameter as a controlled variables.

4.3.2 Multitask BERT Experiment

Later, further experiments were conducted to decrease the learning rate for both pretraining and fine-tuning to determine if the performance of multitask BERT is improved. Specifically, the experiment I conducted decreased the learning rate of pretraining to $1e^{-}5$ and the learning rate of finetuning to $1e^{-}7$. This experiment also decreased the number of epochs of pretraining to 2 epochs and of finetuning to 2 epochs. This helped to decrease the training time as the Quora dataset for paraphrase detection takes significantly longer to run as it is much larger than the SST and SemEval STS datasets.

4.3.3 BERT Extension Experiments

The experiment for cosine similarity fine-tuning utilized the same hyperparameters for the learning rate and epochs in order to study the effect of using CosineEmbeddingLoss for finetuning STS in a controlled setting. My project also experimented with different initial learning rates and number of epochs for the learning rate schedule extension. The initial experiment for using a learning rate schedule only for sentiment analysis involved pretraining for 1 epoch at a learning rate of $1e^{-}3$ and finetuning for 2 epochs at a learning rate of $3e^{-}5$. I chose to try a learning rate of $3e^{-}5$ as that was

one of the learning rates recommended by the BERT paper by Devlin et al. (2019). When applying learning rate schedules to all three downstream tasks, my project used the same initial learning rates as the initial experiment of using a learning rate schedule for sentiment analysis. However, the number of epochs for fine-tuning was decreased from 2 epochs to only 1 epoch. The project involved experiments with contrastive learning using various temperature parameters. A batch size was 8 used, the learning rate was $1e^{-}5$, and 3 training epochs were used.

4.4 Results

Table 1: Test Set Results Summary

Model	SST Acc	STS Acc	Pearson Corr	Overall
BEAKER: Learning Schedule SST	53.5%	36.9%	0.102	0.485

Table 2: Dev Set Results Summary

Model	SST Acc	STS Acc	Pearson Corr	Overall
Multitask BERT	18.6%	37.5%	0.185	0.385
CosineEmbeddingLoss	27.5%	37.5%	0.088	0.398
Learning Schedule for SST	52.9%	37.5%	0.095	0.484
Learning Schedule 3 Tasks	28.6%	43.3%	0.383	0.470

Table 3: Baseline for BERT on Sentiment Analysis

Model	Dev Acc
Pretraining for SST	39.0%
Pretraining for CFIMDB:	78.0%
Finetuning for SST:	51.5%
Finetuning for CFIMDB:	96.6%

The results from this project's experiments demonstrate that using a linear learning rate schedule with linear warmup was quite effective at increasing the performance of BERT from the initial baseline listed as Multitask BERT in Table 2. For instance, accuracy increased from 18.6% to 28.6% for sentiment analysis on SST and from 37.5% to 43.3% on STS. Pearson correlation also increased from 0.185 to 0.383 when using a learning schedule for all three tasks. This increase in performance is likely due to the learning schedule leading the optimizer to gradually move towards the optimum, slowing down as it reaches closer to the minimum. The learning schedule prevents the optimizer from jumping over the optimum due to using too large of a learning rate. Surprisingly, CosineEmbeddingLoss led to a decrease in Pearson correlation for the STS task from 0.185 to 0.088. This was a surprise as I had expected cosine similarity fine-tuning to improve the performance of BERT. However, it is possible that too much fine-tuning could have led to over-fitting of the data, leading to worse performance. For contrastive learning, we find the temperature parameter of 1 to be most effective for the SST task with a dev accuracy of 47.3

5 Analysis

In this analysis, we will examine the prediction outputs for contrastive learning to see how the model made decisions based on certain input sentence pairs. The contrastive learning BERT model returned a predicted similarity score of -0.238 for the two sentences "118 people killed in twin blasts in Nigeria." and "Ten killed in new blast in Russia." Thus, the model was able to recognize the difference in number and location (118 vs Ten and Nigeria vs Russia). Another example of a sentence pair analyzed by the contrastive learning model was the two questions "@Lustig Andrei, So Lyndon Johnson and John Kennedy were compatible?" and "@edgarblythe, So Lyndon Johnson and John Kennedy were compatible?" The predicted similarity score was -0.223 using the contrastive learning model developed in this project. We see from these results that the model heavily weighed

Table 4: Summary of Results for Contrastive Learning with Varied Temperatures

Temperature (T)	**.1**	**0.5**	**1**	**1.25**
Dev SST Accuracy	.314	.390	.473	0.431
Dev Paraphrase Accuracy	.375	.375	.375	.375
Dev STS Accuracy	.041	.010	-0.058	-.006

the difference in name (@Lustig Andrei vs @edgarblythe). Thus, the contrastive learning model is quite successful in differentiating between people, names, places, and numbers when determining the semantic similarity of two sentences. This is probably due to the use of the cosine similarity measure when calculating the cross entropy loss in contrastive learning.

6 Conclusion

This project examined the baseline performance of BERT with pre-trained weights on sentiment analysis using the SST and CFIMDB datasets and investigated the effects of learning rate schedules, contrastive learning, and CosineEmbeddingLoss on three NLP classification tasks: sentiment analysis, paraphrase detection, and semantic textual similarity (STS). We can learn from the findings of this project that a linear learning rate schedule with a linear warmup is effective in improving the performance of BERT on classification tasks. It especially improves the performance on sentiment analysis when the learning rate is only scheduled based on that task's dataset. CosineEmbeddingLoss was found to actually decrease the performance of this version of BERT's performance on STS when used during finetuning, which was a surprising discovery. Finally, contrastive learning was moderately effective in improving BERT's performance on multiple downstream tasks. This project has several limitations, including the fact that the effects of these extensions depends on my initial implementation of the BERT multitask classifier and the architecture I chose to use for making predictions. Thus, this project's results are not generalizable to all BERT classifiers. Additionally, the training dataset for Quora was much larger than the SST and SemEval datasets, and thus it is challenging to understand the effect of having much more data for paraphrase detection than the other tasks. Future research that could be inspired by this project includes experimenting with different learning rate schedules such as square root decay and other non-linear models as well as various different initial learning rates for the optimizer.

References

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. 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.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, pages 1597–1607. PMLR.

Seungtaek Choi, Myeongho Jeong, Hojae Han, and Seung won Hwang. 2022. C2l: Causally contrastive learning for robust text classification.

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*, pages 4171–4186.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2019. Leveraging pre-trained checkpoints for sequence generation tasks. In *Transactions of the Association for Computational Linguistics (TACL)*.

Lakshay Sharma, Laura Graesser, Nikita Nangia, and Utku Evci. 2019. Natural language understanding with the quora question pairs dataset. *arXiv preprint arXiv:1907.01041*.

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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.