Failures of Improving minBERT with Similarity-based Triplet Networks

Stanford CS224N Default Project

Jinpu Cao
Department of Civil and Environmental Engineering
Stanford University
jinpu@stanford.edu

Abstract

The project implemented the backbones of the original Bidirectional Encoder Representations from Transformers (BERT) model and verified it by performing sentence classification. Multiple fine-tuning techniques are tried to obtain robust and generalizable sentence embeddings on three downstream tasks. According to the experiment, measuring cosine similarity between sentences turns out to be an effective fine-tuning approach that can significantly improve the BERT model's performance on semantic textual similarity analysis. However, similarity-based triplet networks can not improve the model's performance as expected.

1 Key Information to include

- Mentor: Christopher Manning

2 Introduction

Bidirectional Encoder Representations from Transformers (BERT) is a transformer-based model that generates contextual word representations (Devlin et al., 2018). The model has been widely used on many natural language processing (NLP) tasks (De Vries et al., 2019). The project implemented the backbone of the original BERT model and verified it by conducting sentiment analysis. fine-tuning trains a pretrained model on a new dataset without from scratch. The process can produce accurate models with smaller datasets and less training time. Multiple fine-tuning techniques are tried to extend the BERT model to create sentence embeddings that can perform well across a wide range of downstream tasks.

According to the research (Reimers and Gurevych, 2019), the similarity between two embeddings can be computed using their cosine similarity. Our experiment results also show importing cosine similarity can be an effective way to improve the embeddings. Furthermore, to make full use of the potential of sentence similarity, inspired by the triple objective function (Reimers and Gurevych, 2019) and multiple negatives ranking loss (Henderson et al., 2017), we proposed an unsupervised similarity-based fine-tuning approach to promote sentence embeddings on the basis of cosine similarity. However, the technique can not boost embeddings as expected from the experiment results.

3 Related Work

The BERT model is frequently used on many NLP tasks, such as sentiment analysis, paraphrase detection, and semantic textual similarity analysis (STS). Fine-tuning is a critical process for applying the model to specific downstream tasks. Ample fine-tuning approaches are developed to improve sentence embeddings. Reimers and Gurevych (2019) presented a modification of the pre-trained BERT network that uses siamese and triplet network structures to derive semantically meaningful

sentence embeddings that can be compared using cosine similarity. The network performs well and efficiently on semantic textual similarity tasks. Based on this, the project uses the cosine-similarity to improve sentence embeddings in the STS task.

Deep learning via triplet networks was first introduced in Hoffer and Ailon (2018) and has since become a widespread technique in metric learning (Yao et al., 2016; Zhuang et al., 2016). Ein-Dor et al. (2018) first used the triplet networks to solve an NLP-related task with weakly-supervised data. The objective of the triplet networks is to make an anchor closer to its positive sentence than its negative sentence. Multiple Negative Ranking Losses presented in Henderson et al. (2017) shared a similar intuition. However, this kind of approach requires particular data preprocessing step. Positive or negative sentence pairs must be prepared for an anchor sentence before training, which is hard to achieve when only the similarity scores of sentence pairs are available. To address this limitation, we proposed an unsupervised similarity-based approach as a variant or extension of the triplet networks.

4 Approach

4.1 minBERT

The transformer-based BERT model used deeply bidirectional word representations and achieved great success on contextual word representations (Devlin et al., 2018). Some critical aspects of the original BERT model, including multi-head self-attention as well as a Transformer layer, are implemented. Multi-head Self-Attention (Vaswani et al., 2017) consists of a scaled-dot product applied across multiple different heads. The input to each head is to a scaled-dot product that consists of queries Q and keys K of dimension d_k, and values V of dimension d_v. The scaled dot-product attention is computed as:

$$Attention(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{1}$$

Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. The Transformer layer of the BERT transformer consists of multi-head attention, followed by an additive and normalization layer with a residual connection, a feed-forward layer, and a final additive and normalization layer with a residual connection.

Sentiment analysis on two datasets was performed by utilizing pre-trained model weights and fine-tuning the outputted embeddings from the model to validate the minBERT model. Adam Optimizer based on Decoupled Weighted Decay Regularization Loshchilov and Hutter (2017), and Adam Kingma and Ba (2014) were implemented for training the sentiment classifier.

4.2 Baseline

The BERT embeddings are utilized to perform three downstream tasks: sentiment analysis, paraphrase detection, and semantic textual similarity. The baseline model is established by making minimal changes to the minBERT model. The BERT embeddings are normalized by a dropout layer and then mapped to the corresponding outputs with a linear layer. It is worth noting that for paraphrase detection and semantic textual similarity analysis, two normalized embeddings (two sentences) in each sample need to be concatenated together before they are fed into the linear layer and their final output should be a logit. The architectures of three baseline models are shown in Fig 1.

4.3 Cosine-Similarity

It is obvious that the outputs of the baseline models for paraphrase detection and semantic textual similarity analysis might be out of range. The paraphrase detection baseline model output can be normalized with a sigmoid function. The output of the semantic textual analysis should be a number measuring the similarity between two sentences (or two-sentence embeddings).

Cosine similarity was utilized to measure the similarity between two sentence embeddings. It can be calculated as follow:

$$cosine\ similarity\ (\vec{u}, \vec{v}) = \frac{\vec{u} \cdot \vec{v}}{\|\vec{u}\|_2 \|\vec{v}\|_2} \tag{2}$$

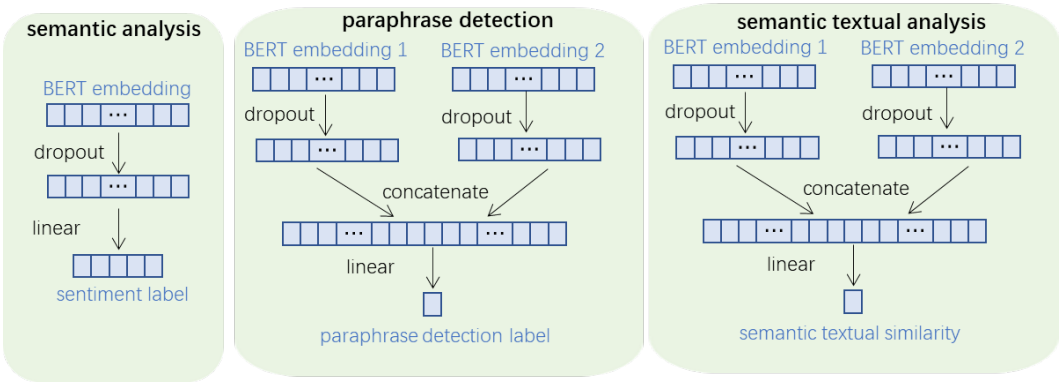

Figure 1: The baseline models for three tasks

where $\vec{u}$, $\vec{v}$ represent two sentence embeddings. The semantic textual similarity analysis model outputs the cosine similarity and is trained with the mean square error loss function. The original cosine similarity ranges from -1 to 1. The output was converted to its absolute value and linearly scaled to the target range to match the range of the similarity used by the dataset. In addition, to update the model's parameters when the BERT model's parameters are frozen, i.e., in the 'pre-train' option, two linear layers are added before the cosine similarity is calculated (the left figure in Fig 2).

4.4 Similarity-based Triplet Networks

Inspired by "Multiple Negative Ranking Loss" (Henderson et al., 2017) and triplet networks (Ein-Dor et al., 2018), we propose an unsupervised similarity-based approach to augment data and improve sentence embeddings. Intuitively, if sentence A is similar to sentence B, given any sentence C, the similarity between A and C should be close to that between B and C. On the other hand, one sentence can not be similar to two sentences that are very different from each other.

The approach can be formalized in the following way, as shown in Fig 2: Given a sample (x_a, x_b, x_c, y_{ab}), where x_a and x_b are a pair of similar sentences whose similarity y_{ab} can be found in the training dataset, x_c is another sentence in the dataset, minimize the following loss function:

$$J(x_a, x_b, x_c, y, \theta) = |S(x_a, x_c) - S(x_b, x_c)| \quad if \quad y_{ab} = S(x_a, x_b) \geq s_0 \tag{3}$$

where θ represents the sentence embeddings and neural network parameters and $S(\cdot)$ is the semantic textual similarity model whose output is cosine similarity between two sentence embeddings. s_0 is the lowest similarity between x_a and x_b. We set $s_0 = 5$ in this work.

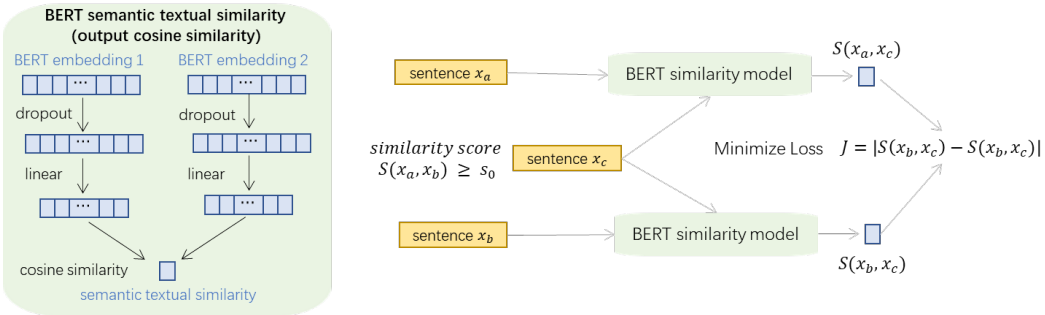

Figure 2: The similarity-based triplet model

It turns out that loss function 3 converges too slowly. The following loss function is used in training the model to speed it up.

$$L_{cd} = \|S(x_c, x_d) - y_{cd}\|_2$$
$$L_{ab} = \|S(x_a, x_b) - y_{ab}\|_2$$
$$L_{abc} = \|S(x_a, x_c) - S(x_b, x_c)\|_2$$

Dataset Name	SST	CFIMDB	Quora	STS
Task	sentiment classification		paraphrase detection	semantic textual similarity analysis
Sample's Input	a sentence	a sentence	a pair of sentences	a pair of sentences
Sample's Output	a label of 5 classes	binary label	binary label	a scale from 0 (unrelated) to 5 (equivalent meaning)
Evaluation Metric	accuracy	accuracy	accuracy	Pearson correlation
Train Size	8,544	1,701	141,506	6,041
Dev Size	1,101	245	20,215	864
Test Size	2,210	488	40,431	1,726

Table 1: Dataset Details

learning rate	epoch	hidden dropout probability	embedding size
pretrain: 1e-3 finetune: 1e-5	10	87.0	768

Table 2: Common model and training configurations on three datasets

$$L_{abd} = \|S(x_a, x_d) - S(x_b, x_d)\|_2$$

$$Loss = \lambda_1 L_{cd} + \lambda_2 L_{ab} + \lambda_3 L_{abc} + \lambda_4 L_{abd} \tag{4}$$

where x_a, x_b and x_a, x_b are two pairs of sentences whose similarity scores (y_{ab}, y_{cd}) are known. Similarly, $y_{ab} = S(x_a, x_b) \geq s_0$. λ is used to scale the weights of each loss component. When $\lambda_1 = 1, \lambda_2 = \lambda_3 = \lambda_4 = 0$, the similarity-based triplet model degrades to the baseline model.

5 Experiments

5.1 Data

Two datasets are used for sentiment classification analysis: Stanford Sentiment Treebank (SST) (Socher et al., 2013) and CFIMDB (Maas et al., 2011). Quora dataset (quo) is used for paraphrase detection. SemEval STS Benchmark dataset (Agirre et al., 2013) is used for semantic textual analysis. The details about these datasets can be retrieved in Table 1.

5.2 Evaluation method

Accuracy is utilized to evaluate the model's performance on sentiment classification and paraphrase detection. The Pearson correlation of the true similarity values against the predicted similarity values is used to evaluate the model's performance on semantic textual analysis.

5.3 Experimental details

The model is trained and tested on Google Colab. The GPU model is NVIDIA A100-SXM4-40GB. The model and training configuration are the same as the default in the handout (as Table 2 shows). Different batch sizes are selected for three tasks. For sentiment classification, the batch size is set to 16, and it takes about 1 minute to fine-tune the model each epoch. For paraphrase detection, the batch size is set to 96, and it takes about 10 minutes to fine-tune the model each epoch since the Quora dataset is much larger than other datasets. For the semantic textual similarity analysis, the batch size is set to 8. It takes 2 minutes and 10 minutes to fine-tune the baseline model and the similarity-based triplet model each epoch, respectively.

	Dev Set	Test Set	Best Model
SST Accuracy	0.520	0.528	Base Model
Paraphrase Accuracy	0.726	0.726	Base Model
STS Correlation	0.713	0.678	Base Model (cosine similarity)
Average Score	0.653	0.644	

Table 3: Results on Three Datasets

5.4 Results

Table 3 shows the best results on validation and testing set for three tasks and their corresponding models. The BERT model gets a 0.653 average score on the validation set and a 0.644 average score on the test set. All these results are obtained from fine-tuning the BERT model for three tasks. The best SST accuracy is achieved by fine-tuning the model on the dataset. However, the best paraphrase accuracy and STS correlation score are achieved by fine-tuning the model for three tasks simultaneously. As expected, the cosine similarity works better than the baseline (Fig. 1). However, the similarity-based triplet model can not improve the model's performance further.

5.4.1 Cosine Similarity

Figure 3 shows the cosine similarity and baseline model results. The figure shows that outputting the cosine similarity can significantly improve sentence embeddings for the semantic textual similarity task. This makes sense because cosine similarity measures the similarity between two vectors. Figure 3 also shows the validation accuracy of the other two tasks. Since we only fine-tuned the BERT model for the STS task, the model performance on the other two tasks did not improve.

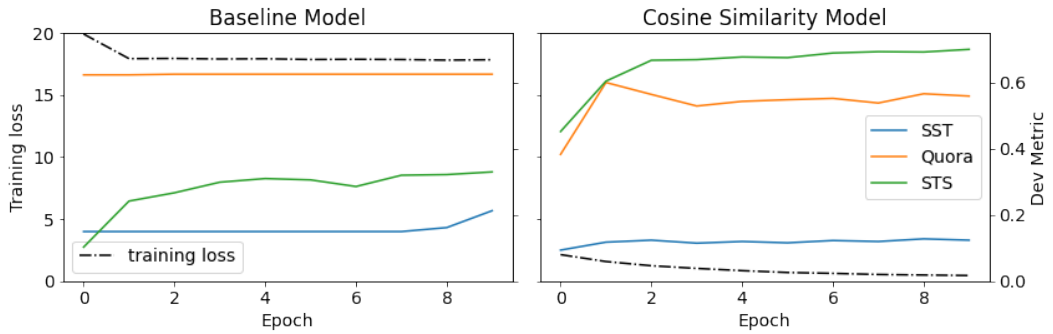

Figure 3: Results of cosine similarity model and baseline model

5.4.2 Similarity-based Triplet Networks

When the similarity-based triplet model was trained, different λ combinations were tried to improve the sentence embeddings in the STS task. Figure 4 shows the correlation score on the validation set of these similarity-based triplet models. Surprisingly, none of these triplet networks outperforms the baseline, which is just outputting the cosine similarity of two sentence embeddings (the left part of Fig 2). Even an ideal model should minimize each part of the loss function (see equation 4). The triplet components, such as L_{abc} and L_{abd}, seem to import too much noise for the model. When we decrease the loss components in equation 4 (by setting the corresponding $\lambda = 0$) step by step, the sentence embeddings will improve accordingly.

We fine-tuned the BERT model on three datasets at the same time, and the results are shown in Figure 4 (right). Model performance on three validation datasets will fluctuate with the model iterating.

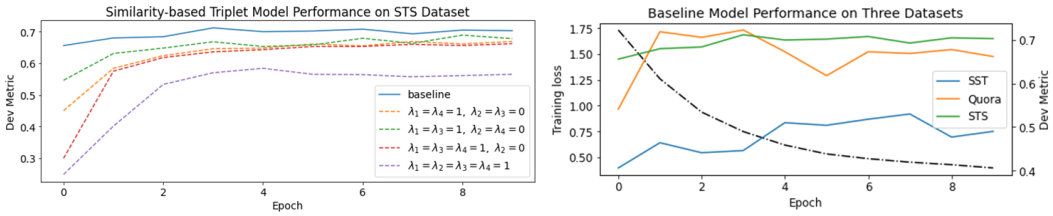

Figure 4: Results of similarity-based triplet model

6 Analysis

According to the research "Multiple Negative Ranking Loss" (Henderson et al., 2017) and triplet networks (Ein-Dor et al., 2018), the similarity-based triplet networks should work theoretically. However, importing these triplets during training does not help improve the sentence embeddings in the semantic textual similarity analysis task. It might result from the following reasons:

1) The number of similar sentence pairs is not larger enough compared to the total number of the STS dataset. In the training set, there are only 283 pairs of similar sentences whose similarity score is 5. When training the triplet networks, we randomly selected a batch of similar sentence pairs from the pool and calculated the corresponding loss function (equation 4) batch by batch. With limited similar sentence pairs, the effect of the triplet loss can not be presented.

2) The weight configurations of each loss component are not proper. Even though multiple λ combinations have been tried, many other choices can be explored. The magnitude of the triplet loss component, such as L_{abc} and L_{abd}, tends to be larger than the baseline loss component, i.e., L_{cd}. So, increasing λ_1 might work.

7 Conclusion

Importing cosine similarity turns out to be an efficient way to boost sentence embeddings on semantic textual similarity tasks, compared to the simple concatenation approach. The proposed similarity-based triplet networks make sense theoretically. However, the approach can not improve the sentence embeddings as expected due to the number of similar sentence pairs.

References

Quora dataset. `https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs`. Accessed: 2023-03-04.

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.

Wietse De Vries, Andreas van Cranenburgh, Arianna Bisazza, Tommaso Caselli, Gertjan van Noord, and Malvina Nissim. 2019. Bertje: A dutch bert model. *arXiv preprint arXiv:1912.09582*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Liat Ein-Dor, Yosi Mass, Alon Halfon, Elad Venezian, Ilya Shnayderman, Ranit Aharonov, and Noam Slonim. 2018. Learning thematic similarity metric using triplet networks. In *Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics (ACL).

Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. *arXiv preprint arXiv:1705.00652*.

Elad Hoffer and Nir Ailon. 2018. Deep metric learning using triplet network.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y 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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Ting Yao, Fuchen Long, Tao Mei, and Yong Rui. 2016. Deep semantic-preserving and ranking-based hashing for image retrieval. In *IJCAI*, volume 1, page 4.

Bohan Zhuang, Guosheng Lin, Chunhua Shen, and Ian Reid. 2016. Fast training of triplet-based deep binary embedding networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5955–5964.