Generalizing BERT through Multi-Task Learning

Stanford CS224N Default Project

Caroline Wang
Department of Computer Science
Stanford University
carol23@stanford.edu

Abstract

Bidirectional Encoder Representations from Transformers or BERT is a model that uses word representations to perform tasks such as sentiment analysis, paraphrase detection, and semantic textual similarity. In this paper, a basic BERT model is implemented as a baseline model. Then, in an attempt to improve BERT's ability to generalize well across the three downstream tasks listed above, this paper implements a multi-task method of training BERT, and it uses cosine similarity to identify semantic textual similarity. The resulting accuracy and correlation scores gained from applying BERT to sentiment analysis, paraphrase detection, and semantic textual similarity tasks show that these methods are effective.

1 Key Information to include

- Mentor: Gabriel Poesia
- External Collaborators (if you have any): N/A
- Sharing project: N/A

2 Introduction

Creating a BERT model that can generalize well across several downstream tasks saves time on implementing separate models for separate tasks. This paper discusses a multi-task BERT model that aims to perform well on sentiment analysis, paraphrase detection, and semantic textual similarity.

Sentiment analysis classifies a text's polarity, that is, whether the text contains negative, positive, or neutral feelings. Paraphrase detection determines whether or not two texts are paraphrases of each other. Semantic textual similarity rates on a scale the similarity between two texts.

In order to create a BERT model that performs well on all three of the tasks, multi-task learning is implemented as outlined by Bi et al. (2022). The BERT model is trained simultaneously on all three tasks, where the loss functions from each task are summed together. This improved the accuracy and similarity scores across all three tasks.

On top of multi-task learning, cosine similarity is then used for semantic textual similarity according to Reimers and Gurevych (2019), to see if the model could be improved further. In this implementation, word embeddings were compared via cosine similarity, where similar embeddings had a cosine similarity score of one, and non similar embeddings had a score of zero. This model was also trained with multi-task learning. This further improved the accuracy and similarity scores across all three tasks.

3 Related Work

As mentioned previously, this paper draws on the multi-task learning implementation from Bi et al. (2022). In their paper, the main goal was to improve news recommendation systems by looking at titles of news articles. To supplement their BERT model, they implemented multi-task learning where the model learned auxiliary features, such as category classification and named entity recognition, in order to improve the main task of encoding news titles. In contrast with their paper, this implementation treats all three tasks as main tasks.

Using cosine similarity for semantic textual similarity draws from Reimers and Gurevych (2019). While their main goal was to recreate a BERT model using siamese and triplet networks to create sentence embeddings, their method of using cosine similarity is relevant to this BERT model's implementation.

4 Approach

4.1 Baseline

The baseline BERT model details can be found in the BERT handout. As a quick summary of how this model obtains its sentence embeddings, it first converts sentences into tokens, applies a learnable embedding layer consisting of token embeddings and position embeddings, and finally applies a transformer layer. The transformer layer is composed of multi-head attention, an additive and normalization layer with a residual connection, a feed-forward layer, and finally another additive and normalization layer with a residual connection. The sentence embeddings are then used to predict sentiment analysis, paraphrase detection, and semantic textual similarity.

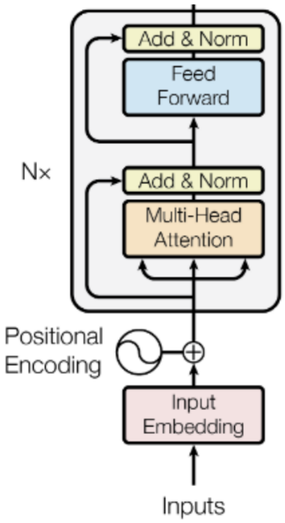

Figure 1: Encoder Layer of Transformer used in BERT (Vaswani et al., 2017)

The baseline BERT model has pretrained weights from the original BERT model (Devlin et al., 2018). The model uses these pretrained weights and sentence embeddings obtained from datasets corresponding to sentiment analysis, paraphrase detection, and semantic textual similarity, to predict these three tasks.

In this paper's baseline BERT model, to detect sentiment analysis, a linear layer is applied to the word embeddings and the cross entropy loss function is optimized for. For both paraphrase detection and semantic textual similarity, the two word embeddings that are going to be compared are first combined and then passed through a linear layer. Paraphrase detection optimizes the cross entropy loss whereas semantic textual similarity optimizes the mean squared error loss.

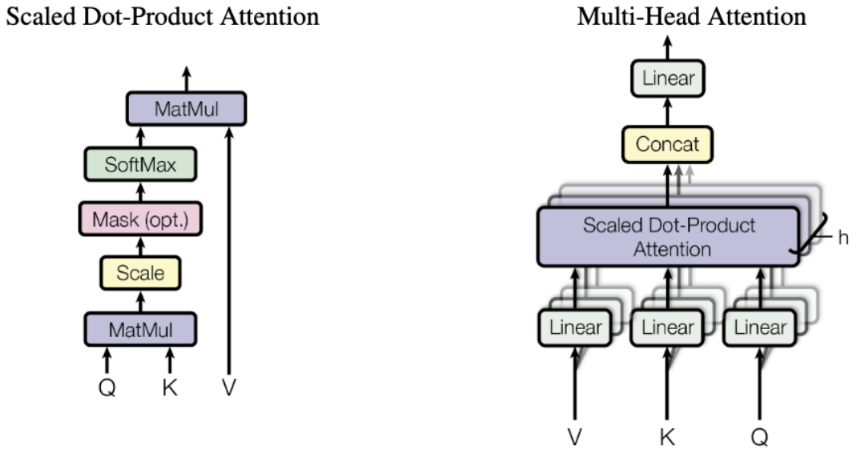

Figure 2: Scaled Dot-Product and Multi-Head Self Attention (Vaswani et al., 2017)

4.2 Multi-task Learning Model

The multitask learning model further updates the already pretrained weights to optimize for the loss functions. The loss functions from the three different tasks are simply added together as shown:

$$L = L_{sentimentanalysis} + L_{paraphrasedetection} + L_{semantictextualsimilarity} \tag{1}$$

4.3 Cosine-Similarity Model

The cosine similarity model also uses multitask learning. For predicting semantic textual similarity, rather than combining the two word embeddings, the embeddings are first passed through a linear layer and then their cosine similarity score is computed. The mean squared error loss is still optimized for. The cosine similarity is computed as shown:

$$similarity = \frac{x_1 \cdot x_2}{max(||x_1||_2 \cdot ||x_2||_2, \epsilon)} \tag{2}$$

where $\epsilon = 1e - 08$

5 Experiments

5.1 Data

The Stanford Sentiment Treebank (SST) Dataset is made up of 11,855 sentences from movie reviews, and has been parsed by the Stanford parser, resulting in 215,154 unique phrases (Socher et al., 2013). Each phrase has been labeled as either negative, somewhat negative, neutral, somewhat positive, or positive by three human judges. This dataset will be used to evaluate the BERT model's sentiment classification accuracy. The data is split as followed:

- train (8,544 examples)
- dev (1,101 examples)
- test (2,210 examples)

The Quora dataset contains 400,000 question pairs with binary labels indicating if the questions are paraphrases of each other (Fernando and Stevenson, 2008). This dataset will be used to evaluate the BERT model's paraphrase detection accuracy. The data is split as followed:

- train (141,506 examples)

- dev (20,215 examples)
- test (40,431 examples)

The SemEval STS Benchmark dataset contains 8,628 sentence pairs and labels indicating their similarity, with 0 meaning the sentences are unrelated and 5 meaning the sentences are equivalent (Agirre et al., 2013). This dataset will be used to evaluate the BERT model's ability to detect semantic textual similarity. The Pearson correlation between predicted and true scores will be calculated. The data is split as followed:

- train (6,041 examples)
- dev (864 examples)
- test (1,726 examples)

5.2 Evaluation method

The models will be compared via their dev accuracy and Pearson correlation scores, and the cosine similarity model will also be evaluated by its test accuracy and test Pearson correlation scores. The overall dev and test scores will be computed by averaging the scores from the three tasks.

5.3 Experimental details

The baseline BERT model was trained with the "pretrain" flag with epochs=5, learning rate=1e-3, dropout rate=0.3, and batch size=8.

Both multi-task and cosine similarity models were trained with the "finetune" flag with epochs=3, learning rate=1e-5, dropout rate=0.3, and batch size=8.

5.4 Results

Model	Overall dev score	SST dev Acc	Paraphrase dev Acc	STS dev Corr
Baseline	0.439	0.379	0.664	0.274
Multi-task Learning	0.529	0.511	0.734	0.342
Cosine Similarity	0.566	0.514	0.757	0.427

Table 1: Comparing dev scores across models when tested on the SST, Quora, and STS datasets.

Model	Overall test score	SST test Acc	Paraphrase test Acc	STS test Corr
Cosine Similarity	0.569	0.542	0.758	0.406

Table 2: The cosine similarity model's test scores when tested on the SST, Quora, and STS datasets.

Implementing multi-task learning improved the scores across all three tasks, since BERT parameters could be updated according to all of the word embeddings from the different datasets. Implementing cosine similarity further improved the dev scores for all three tasks, although it improved the STS dev correlation more significantly than for the SST and paraphrase dev accuracy. This is because cosine similarity was only used in the semantic textual similarity task. Since the SST and paraphrase accuracies did not decrease, the loss functions seem to be well aligned with each other.

6 Analysis

When looking at how the cosine similarity implementation affects STS correlation scores, note that there are many significant flaws. In the dev results, the model tends to predict very low similarity scores, regardless of whether or not the sentences are actually similar.

Examples:

A family is playing on the beach with their dog.
A naked little girl being thrown into the air.
Model's similarity score: 0.53
Actual score: 0.0

Boston bombing suspect buried in Virginia.
Boston bomb suspect buried in Virginia cemetery.
Model's similarity score: 1.79
Actual score: 5.0

The model seems to only give high similarity scores when the sentences are long and contain a lot of identical words.

Examples:

On Monday, as first reported by CNET News.com, the RIAA withdrew a DMCA notice to Penn State University's astronomy and astrophysics department.
Last Thursday, the RIAA sent a stiff copyright warning to Penn State's department of astronomy and astrophysics.
Model's similarity score: 2.95
Actual score: 1.75

""This has been a persistent problem that has not been solved,"" investigation board member Steven Wallace said.
""This was a persistent problem which has not been solved, mechanically and physically,"" said board member Steven Wallace.
Model's similarity score: 3.68
Actual score: 4.0

Thus, when given long sentences with identical words, the cosine similarity model predicts a higher similarity score, but for other types of sentences, the model gives a low similarity score. For two short sentences that are not similar, and for two long sentences that are similar, the model will have a better chance at predicting their similarities.

The cosine similarity model does not seem to be able to detect words that humans might identify as similar.

Example:

A boy is playing a key-board.
A boy is sitting in a room playing a piano by lamp light.
Model's similarity score: 0.43
Actual score: 3.25

In this example, the model might not recognize that "key-board" and "piano" are related.

7 Conclusion

Using multi-task learning and using cosine similarity as a method to identify word embedding similarity allowed the BERT model to generalize well across the three tasks of sentiment analysis, paraphrase detection, and semantic textual similarity. There were limitations to this model, namely a lack of time and resources to complete more epochs of training and experiment in finding hyperparameters and adding layers that could have improved the performance. For future work, implementing gradient surgery as a way to combine loss functions such that their gradient descents do not conflict with each other would be a good next step (Yu et al., 2020). Trying different loss functions, such as the cosine embedding loss for semantic textual similarity and the multiple negatives loss function for paraphrase detection could also improve performance(Henderson et al., 2017).

References

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, Atlanta, Georgia, USA. Association for Computational Linguistics.

Qiwei Bi, Jian Li, Lifeng Shang, Xin Jiang, Qun Liu, and Hanfang Yang. 2022. MTRec: Multi-task learning over BERT for news recommendation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2663–2669, Dublin, Ireland. Association for Computational Linguistics.

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.

Samuel Fernando and Mark Stevenson. 2008. A semantic similarity approach to paraphrase detection.

Matthew L. 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. *CoRR*, abs/1705.00652.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *CoRR*, abs/1908.10084.

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, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 5824–5836. Curran Associates, Inc.