ExtraBERT: Applying BERT to Multiple Downstream Language Tasks

Stanford CS224N Default Project

Rishi Dange
Department of Computer Science
Stanford University
rdange@stanford.edu

Isaac Gorelik
Department of Computer Science
Stanford University
gorelik@stanford.edu

Abstract

In this project, we implement the classic BERT architecture and demonstrate its applicability across three distinct natural language processing tasks: sentiment analysis, paraphrase detection, and semantic textual similarity analysis. We begin with an architecture fine-tuned just for sentiment analysis, and proceed via a set of strategic iterations to ultimately arrive at a multitask learning model that could theoretically be generalized to even more tasks. We find that incorporating additional fine-tuning data (specifically, the Stanford Natural Language Inference corpus) and thoughtfully increasing the complexity of our downstream predictor functions provide the most positive impact to our model's performance and ability to generalize on unseen data. We also implement a version of gradient projection in order to mitigate gradient conflicts among the separate tasks; our results show improvement on paraphrase detection and semantic textual similarity analysis but not on sentiment analysis. Nevertheless, this project demonstrates the applicability and adaptability of the BERT framework that and sets the stage for more complex work to come.

1 Key Information

- **Mentor:** Arvind Mahankali
- **Team Contributions:** Work was split evenly, as both partners worked on the design, implementation, and writeup. For project extensions, Isaac focused on SNLI, and Rishi focused on gradient surgery.

2 Introduction

The overarching goal of this work is to implement and enhance the Bidirectional Encoder Representations from Transformers (BERT) model, leveraging its multi-head self-attention and transformer layers as outlined by (Devlin et al., 2018) in the original BERT paper. Our focus is on exploring the efficacy of the BERT model across three fundamental Natural Language Processing (NLP) tasks: sentiment analysis, paraphrase detection, and semantic textual similarity analysis. Moreover, the project aims to innovate upon the baseline BERT model through various strategies to augment its performance on these downstream tasks, thereby pushing the boundaries of what is achievable with current deep learning techniques in NLP.

More specifically, the three canonical NLP tasks that we target are:

1. Sentiment Analysis: Determining the sentiment expressed in a piece of text, categorizing it as positive, negative, or neutral.
2. Paraphrase Detection: Identifying if two texts convey the same meaning, an essential task for ensuring the uniqueness of textual content and understanding linguistic variations.

3. Semantic Textual Similarity Analysis: Assessing the degree of semantic similarity between two sentences, which is crucial for tasks such as information retrieval, question answering, and summarization.

For our basic training, development, and testing data, we utilize the Stanford Sentiment Treebank (SST) dataset along with the CFIMDB movie review dataset for the purpose of sentiment analysis, a Quora labeled paraphrase dataset for the purpose of paraphrase detection, and the SemEval STS Benchmark dataset for the purpose of semantic similarity analysis.

The foundation of our project is the BERT model, where the key neural methods in the original architecture are bidirectional encoders, multi-head self-attention, and transformer layers. From there, we proceed in an iterative improvement process toward our overarching multitasking goal that highlights the benefits and drawbacks of each strategy along the way, using the performance of our basic BERT model on sentiment analysis as a baseline for model performance. These model improvement strategies include adding complexity (in terms of algorithms, features, and layers) to the downstream task prediction functions, incorporating additional training data for the fine-tuning of the semantic similarity task (specifically, we use the Stanford Natural Language Inference developed by Bowman et al. (2015) as a means of better connecting pairs of sentences during the fine-tuning process), and implementing multi-task learning with gradient projection to avoid gradient conflicts.

For sentiment analysis and paraphrase detection, we use accuracy (correct classifications as a fraction of total classifications) to evaluate our model's performance. For semantic textual similarity analysis, we use the Pearson correlation of the true similarity values against the predicted similarity values, as used by Agirre et al. (2013) in the original SemEval paper. We find that our overall performance on the three tasks increases with the additional prediction function complexity, and we also find that the incorporation of the SNLI dataset generates a considerable improvement for our semantic similarity task (critically, it also does not appear to hinder the other tasks' performances). However, our gradient projection implementation hinders performance on the sentiment analysis task. Nevertheless, we continue to believe in its potential and as future work would spend time adjusting our loss functions prior to performing gradient projections in order to attempt to avoid this hindrance.

3 Related Work

Our project aims to build and expand upon the original Bidirectional Encoder Representations from Transformers (BERT) model as presented by (Devlin et al., 2018) to build a model capable of multi-tasking in the three downstream tasks of sentiment analysis, paraphrase detection, and similarity detection. As such, we will first introduce BERT and then discuss the relevant research for approaching the downstream multi-tasking:

Recently, Natural Language Processing (NLP) has experienced a paradigm shift with the advent of models like BERT, which rely on the self-attention mechanism inherent in the transformer architecture circumvent the need for large, sequential sequence to sequence models such as RNNs and LSTMs. The default BERT model used was pre-training on a large corpus of text using two tasks—masked language modeling and next sentence prediction tasks to learn contextual representations of words and sentences. We relied on this baseline BERT model, and incorporated additional output layers built on top of pooled sentence outputs to perform a wide array of NLP tasks, leveraging its bidirectional training to understand the full context of a word based on its surroundings.

Gradient-based optimization techniques have been crucial in fine-tuning deep learning models. The concept of gradient surgery, as explored by (Bi et al., 2022), involves modifying gradient vectors to enhance the training of multi-task models, and has demonstrated that careful management of gradient updates can lead to improved generalizability across tasks. By integrating gradient projection methods, we aim to mitigate the adverse effects of competing task gradients, a common challenge in multi-task learning scenarios.

Feature engineering plays a pivotal role in the effectiveness of NLP models. For paraphrase detection and semantic textual similarity analysis, our project employs cosine similarity and L1 distance measures between sentence vectors, as used by (Reimers and Gurevych, 2019) in a siamese appraoch, where input sentences were passed through two identical BERT models. This mirrors our approach, where we feed inpt sentences through the same BERT model for pairwise tasks such as paraphrase detection and similarity assessment. By fusing these similarity metrics with BERT's deep contextualized representations, we intend to create a more nuanced and precise model capable of discerning intricate linguistic relationships.

In harnessing these advancements, our project seeks to address existing limitations in the application of BERT to NLP tasks. Through exploring gradient techniques and specialized neural architecture geared towards three distinct donwnstream tasks, we make a compelling case for the adaptability and growth potential of using transformer based models like BERT in a multi-faceted NLP landscape.

4 Approach

We begin our iterative approach toward a successful multitask classifier with our basic BERT model that is trained to perform sentiment classification on the Stanford Sentiment Treebank (SST) dataset. Toward our goal of producing rich sentence embeddings that can perform all three of our desired downstream tasks, we proceed with the following phases.

Phase 1. Linear predictors using sentiment training

Our baseline BERT model is trained solely on the sentiment classification task using the corresponding dataset. In that light, our very first pass at predicting sentiment, detecting paraphrases, and analysis semantic similarity comes in the form of a simple downstream linear layer for each task that transforms the input dimensionality into the desired output dimension (and into the desired task output range).

Phase 2. Basic multitasking during training

In order to allow our model to perform well at all three tasks, we want to incorporate all three tasks in our training process. In that light, for each epoch, we first proceed through all training examples for the sentiment task, calling `loss.backward()` each time. We then proceed through the paraphrasing examples, finally ending with the semantic textual similarity examples. In this manner, our model learns its parameters based on all three tasks rather than just sentiment analysis.

Phase 3. More complex predictor functions

In the previous phases, our downstream predictor functions simply take input sentence embeddings and use linear layers to produce output values. We now enrich these functions with additional layers and features.

For our sentiment predictor, we utilize a dropout layer followed by a fully-connected layer in order to aid in the process of generalizing to unseen data. Then, we use a batch normalization layer to add stability to the learning process, which is followed by a nonlinearity (ReLU). After, we use four different versions of the same layers along with a final dropout layer before passing the resulting values to our output linear layer.

For our paraphrase predictor, we begin with the input sentence embeddings for the two (batches of) sentences. To enhance our model's generalizability and prediction strength, we apply a dropout and a linear layer on those embeddings before computing their cosine similarity. As features, we take the two original batches, the element-wise absolute difference between the two original batches (Reimers and Gurevych, 2019), and the modified embedding cosine similarity scaled so that it is between -9 and 1 (experimentally derived during our hyperparameter tuning process in order to make the paraphrase threshold sufficiently stringent). Following a dropout layer, a fully-connected layer, and a nonlinearity (ReLU), we can finally utilize our output linear layer.

Lastly, for our semantic similarity predictor, we (as with paraphrase detection) begin with the input sentence pair embeddings and apply a dropout and a linear layer on those embeddings before computing their cosine similarity. As features, we take the two original batches, the element-wise absolute difference between the two original batches (Reimers and Gurevych, 2019), and the element-wise absolute value of the cosine similarity between the modified emeddings. We take the absolute value for each element in the tensor because two opposite sentences (a cosine similarity of -1) should pertain to the same topic more than two orthogonal sentences. Following a dropout layer, a fully-connected layer, and a nonlinearity (ReLU), we can finally utilize our output linear layer (after which we use a sigmoid function and scale to achieve our desired output range).

Phase 4. Additional SNLI data for fine-tuning

In this phase, our goal is to leverage extra data sources during the training process to fine-tune the model, thereby enhancing its ability to generalize to unseen data. We specifically use the Stanford Natural Language Inference corpus developed by Bowman et al. (2015) as a means of better connecting pairs of sentences during the fine-tuning process. The dataset consists of pairs of sentences with labels "entailment," "contradiction," and "neutral." We transform this dataset and its labels into a corresponding dataset applicable to our semantic textual similarity analysis task, selecting the semantic similarity labels based on our manual inspection of the corpus as well as the reults of our hyperparameter tuning process. During the fine-tuning process, we intend for this corpus to provide important information for training and parameter adjustment.

Phase 5. Interleaved batches across tasks

As defined in Phase 2, our basic multitasking approach works with our three tasks one at a time. To enhance our multitasking so that the tasks are learned concurrently, we define an overall loss function during training

$$\mathcal{L}_{\text{TOT}} = \mathcal{L}_{\text{sentiment}} + \mathcal{L}_{\text{paraphrase}} + \mathcal{L}_{\text{similarity}} \tag{1}$$

that represents the total loss across all three tasks. During each epoch, the batches of examples across tasks are interleaved so that parameter updates are made based on this new total loss function, hence incorporating all three tasks rather than solely sentiment analysis. In the case of unequal batch sizes, once a particular task has run out of its examples, only the remaining tasks continue providing their examples for training.

Phase 6. Gradient-projected multitasking

In order to further enhance the abilities of this multi-task learning strategy, we follow the 2022 MTRec work to employ a modified version of the Gradient Surgery technique (Bi et al., 2022) that, in order to deal with gradient conflicts, generally projects the gradient of a particular task $\mathbf{g}_i$ onto the normal plane of the conflicting gradient $\mathbf{g}_j$ using the equation

$$\mathbf{g}_i = \mathbf{g}_i - \frac{(\mathbf{g}_j \cdot \mathbf{g}_i)}{||\mathbf{g}_j||^2} \cdot \mathbf{g}_j. \tag{2}$$

Specifically, we define sentiment analysis as our "main" task and define paraphrase detection and semantic textual similarity analysis as our "auxiliary" tasks. For a given auxiliary task gradient $\mathbf{g}_{\text{aux}} \in \{\mathbf{g}_{\text{paraphrase}}, \mathbf{g}_{\text{similarity}}\}$, we project the gradient $\mathbf{g}_{\text{aux}}$ onto the normal plane of $\mathbf{g}_{\text{main}} = \mathbf{g}_{\text{sentiment}}$ if and only if they conflict (i.e., their dot product is negative). If the gradients do not conflict, we leave the auxiliary gradient unchanged. Our overall gradient used for parameter updates is then

$$\mathbf{g}_{\text{all}} = \frac{\mathbf{g}_{\text{main}} + \mathbf{g}^{\text{proj}}_{\text{paraphrase}} + \mathbf{g}^{\text{proj}}_{\text{similarity}}}{3}, \tag{3}$$

where $\mathbf{g}^{\text{proj}}_{\text{paraphrase}}$ and $\mathbf{g}^{\text{proj}}_{\text{similarity}}$ represent the (potentially) projected auxiliary gradients.

5 Experiments

5.1 Data

We utilize the following datasets for training, development, and testing.

1. **Task: Sentiment Analysis**
   Stanford Sentiment Treebank (SST) Dataset: This dataset consists of sentences extracted from movie reviews for the purpose of sentiment analysis. There are 8,544 training examples, 1,101 examples in the dev set, and 2,210 examples in the heldout test set.
   CFIMBD Dataset: This dataset consists of extremely polar movie reviews. This dataset contains 1,701 training examples, 245 examples in the dev set, and 488 heldout test examples.
2. **Task: Paraphrase Detection**
   Quora Dataset: A collection of question pairs from the Quora platform, where each pair is labeled as being a paraphrase (similar meaning) or not. This dataset will primarily support the paraphrase detection task. There are 141,506 training examples, 20,215 examples in the dev set, and 40,431 examples in the heldout test set.

3. **Task: Semantic Textual Similarity Analysis**
   SemEval STS Benchmark Dataset: A standard dataset for evaluating semantic textual similarity, consisting of sentence pairs annotated with similarity scores. It facilitates the training and evaluation of models on their ability to discern semantic similarities between sentences.

5.2 Evaluation method

Sentiment analysis and paraphrase detection are inherently classification tasks (paraphrase detection being a binary classification task). Consequently, for sentiment analysis and paraphrase detection, we utilize accuracy to evaluate our model's performance:

$$\frac{\textbf{correct classifications}}{\textbf{total classifications}}$$

For semantic textual similarity, we utilize the Pearson correlation of the true similarity values against the predicted similarity values as an indicator of our model's performance, as used by Agirre et al. (2013) in the original SemEval paper. This metric is defined as:

$$\rho_{Y,\hat{Y}} = \frac{\textbf{cov}(Y,\hat{Y})}{\sigma_Y \sigma_{\hat{Y}}}$$

.

5.3 Results

	Overall Score	Sentiment acc.	Paraphrase acc.	STS corr.
P2: Pretr. (10 epochs, 1e-3 LR)	0.438	0.406	0.383	-0.009
P2: Finetun. (10 epochs, 1e-5 LR)	0.511	0.527	0.490	0.030
P3: Finetun. (7 epochs, 5e-6 LR)	0.740	0.484	0.864	0.741
P4: Finetun. (7 epochs, 1e-5 LR)	0.749	0.492	0.860	0.789
P6: Finetun. (7 epochs, 1e-5 LR)	0.577	0.174	0.740	0.632

The above table displays our results on selected phases' pretraining and finetuning runs (where pretraining refers to runs in which the BERT parameters are frozen, and finetuning refers to runs in which those parameters are learned along with the model layer parameters), specifically displaying scores on the development set. Note that there is no row for Phase 5 in the table because we did not complete a full run through of the phase given that it was an intermediary phase for the upcoming Phase 6. Instead, given our time constraint, we sped up our development process by training on a smaller subset of the training data and received development scores also on a subset of the development set, so those runs are not directly comparable to the ones shown in the table. Ultimately, our most accurate model (from Phase 4) achieved a sentiment analysis accuracy of 0.510, paraphrase accuracy of 0.859, and Pearson correlation score (for the similarity task) of 0.752 on the heldout test set.

6 Analysis

On the development set, our model achieved a sentiment accuracy of 0.492, a paraphrase detection accuracy of 0.860, and a SST Pearson correlation of 0.789. We investigated the predictions for each of these datasets, and observed the following interesting patterns with the model's behavior:

6.1 Similarity Analysis

When choosing sentence pairs at random, our model's predictions were quite strong, and were largely within 0.2 of the true similarity score. Overall, our model performed the weakest for sentence pairs of very high similarities (close to 5.0).

In fact, for the 865 points in the dev set, not a single prediction was above 4.3, suggesting that our model seems to have difficulty assigning high similarity scores. Our model does not seem to have as much difficulty on the lower edge.

For instance, the sentence pair "A cat opens a drawer and climbs inside," and "A cat is opening a drawer and climbing inside," should have a similarity score of 5.0, but our model assigns a value of 4.2

6.2 Sentiment Analysis

Somewhat like the similarity predictor, our sentiment classifier seems to perform rather well for movie reviews with “neutral” sentiment, towards the center (i.e. values of 1, 2, or 3). Interestingly enough, we found these neutral reviews rather difficult to discern from one another — although we did not analyse our human classification accuracies on a random batch of reviews, we believe that it is possible that the model might even outperform us for these reviews in the mid-range. However, for very negative reviews, our model was too optimistic, and very often failed to identify the negative reviews as being a 0.

Some examples are:

- "When the film ended , I felt tired and drained and wanted to lie on my own deathbed for a while ." A human reviewer should likely correctly classify this review as a resounding 0, but our model assigned it a sentiment of 1.
- "Expect the same-old , lame-old slasher nonsense , just with different scenery." was also incorrectly classified as 1 rather than 0.

6.3 Paraphrase Analysis

With an accuracy of 0.860, our model seemed to perform rather well for paraphrase detection, but seemed to fall victim to a certain false positive trap. These two pairs of sentences were both classified as paraphrases:

- What might be some reasons that there are more answers than questions on Quora? Are there more answers than questions on Quora?
- Why is Frederiksberg not a part of Copenhagen? Is Frederiksberg not a part of Copenhagen?

It appears that our model gets tricked by situations where one sentence is a yes/no question, and the other is asking to explain the phenomenon (using extremely similar, or even identical, language patterns).

7 Conclusion and Future Work

This work sought to explore the efficacy of the BERT model in tackling the natural language processing tasks of sentiment analysis, paraphrase detection, and semantic textual similarity analysis. During our six phases of modification, we augmented our model in order to make it more suitable for the tasks at hand and to specifically enable it to concurrently learn embeddings pertinent to the three different tasks. Based on our results, we see the adaptability of the BERT model across a variety of natural language processing tasks when provided with the right model setup and modifications. Moreover, we see the importance behind presenting additional fine-tuning data and defining complex yet generalizable predictor functions. While our results for Phase 6 were not as promising as we would have liked on the sentiment analysis task (perhaps due to the limited complexity behind the gradient combination mechanism), its success on the other two tasks make the gradient projection route an intriguing path for future work.

As future work, we would be keen to work on the following aspects of our model.

1. We would like to improve the gradient-projected multitasking functionality of our model. Specifically, we hypothesize that unnormalized loss functions across the tasks combined with a very elementary averaging scheme may be the cause of our lacking performance for sentiment analysis. We would be very interested in exploring this route as a means of further improving our model.
2. We would seek to better tune the hyperparameters of our model for their respective tasks.

3. We would also seek to construct even stronger predictor functions for the three tasks, as better predictors very clearly positively impacted our model's performance.
4. We would be curious to generalize this work to a platform that would allow an end user to multitask across a wide variety of novel tasks given a simple dataset along with labels.

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.

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.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 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. *arXiv preprint arXiv:1810.04805*.

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