minBERT: Contrastive Learning Method

Stanford CS224N Default Project

Long Pham
Department of Computer Science
Stanford University
Ldpham00@stanford.edu

Abstract

This project is split into two components: the first part is to implement a contrastive learning framework to minBERT, and the second part is to further fine-tune the model in an attempt to improve its accuracy. Our hopes in doing so is to improve the performance of the model on three mainstream tasks: sentiment analysis, paraphrase detection, and semantic textual similarity. Our method includes using pre-existing training data that contains positive and negative pairs to train our data and use their cosine similarity as a determinant for the magnitude to which we should "pull in" or "push away" the feature embeddings of the pairs of sentences. We then compare the performance of this new model with our implemented minBERT model in order to evaluate its effectiveness. Our results show that the integration of contrastive learning improve our accuracies for all three downstream tasks. This study demonstrates the effectiveness of the contrastive learning method in the domain of natural language processing and shows promising avenues for future unsupervised implementation in NLP models but also for other domains including computer vision.

1 Key Information to include

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

2 Introduction

Contrastive learning is a machine learning technique in which the model is trained to be able to distinguish between similar and dissimilar pairs of input data. The model does so by "pulling in" pairs of features that are similarly related and "pushing away" pairs of features which are not by comparing their cosine similarity and augments it by a "temperature" value. An exciting implication of this is that this method allows for the model to be trained on a smaller set of data. This is useful in applications where there is not a lot of training data and can also be utilized in unsupervised machine learning.

Contrastive learning has found many uses in a wide range of domains of machine learning including natural language processing and computer vision. However, one difficulty that arises with the implementation of contrastive learning in an NLP model is that language processing is inherently discrete, and so it is more difficult to create positive and negative pairs. Whereas with an image, it can be flipped or grayscaled to produce a new positive pair, it is much more difficult with a sentence. While we can randomly omit a word or replace it with its synonym, doing so does not always guarantee a coherent new sentence nor does it always even guarantee that the new sentence will even have a contextually similar meaning.

The goal of this project is to adapt this idea of contrastive learning into our architecture in hopes of improving the original performance. We aim to use pre-existing labels that already include labels as to whether two sentences are similar, and then applying the contrastive learning loss function to each network architecture. To do so, we will incorporate a cosine similarity function along with cross entropy loss into our original model. We will then continue to finetune the model in hopes of improving its performance.

3 Related Work

Our study was largely inspired by the paper SimCSE: Simple Contrastive Learning of Sentece Embeddings, in which the authors aimed to further improve on current sentence embedding methods and to show the effectiveness of contrastive learning when it is coupled with pre-trained language models including BERT and RoBERTA (Gao et al., 2021). Our experiment adapted the suggestions from this study and applied it to our model for minBERT.

In the study, the authors addressed the limitations of existing approach and are motivated by the potential of contrastive learning. Furthermore, they hope to discover methods that can lead to better sentence embeddings generation and its application to improve the performance of NLP systems in tasks such as semantic textual similarity analysis and natural language inference.

In essence, contrastive learning pulling semantically related neighbors close together and push non-semantically related neighbors apart. In training the model, the authors encoded the input sentences using a pre-trained language model and then fine-tuning all of the parameters using the contrastive learning equation:

$$\ell_i = -log \frac{e^{sim(h_i, h_i^+)/\tau}}{\sum_{j=1}^{N} e^{sim(h_i, h_j^+)/\tau}}$$

where h_i and h_i^+ denote the representation of two semantically related examples, τ is a temperature hyperparameter, and $sim(h_1, h_2)$ is the cosine similarity between the two representations (Gao et al., 2021). By using the same loss function as described above into our minBERT model, we hope to achieve a similar result.

4 Approach

For the baseline, we used the performance of the minBERT model with the Adam Optimizer as outlined in the CS 224N Default Final Project handout (CS224N, 2024). As stated in **Section 5.2** of the handout, for **multitask_classifier.MultitaskBERT.predict_sentiment**, as a baseline, we will call the **forward()** method followed by a dropout and linear layer. Similarly, as a baseline for **multitask_classifier.MultitaskBERT.predict_paraphrase** and **multitask_classifier.MultitaskBERT.predict_similarity**, we will call the **forward()** method followed by a dropout layer on both input sentences. We will then sum the two outputs and apply a linear layer. All of the linear layers take in inputs of size *in_features=config.hidden_size*. The linear layer used for sentiment analysis have outputs of size *out_features=N_SENTIMENT_CLASSES*, which is 5 for our dataset, whereas the linear layers used for paraphrase detection and textual similarity have outputs of size *out_features=1*. We will compare the Pearson correlation coefficients of all three downstream tasks of our finalized model to the scores from this model in order to determine our improvements. A better Pearson correlation coefficient will show us that the techniques we implemented are beneficial to the model.

To improve this, one thing we did was to augment our neural network architecture. The most notable changes are made to the architecture of **predict_paraphrase()** and **predict_similarity()** in which rather than summing the two layers and pipelining the tensor into a linear layer, the two embeddings are each processed into their own separate linear layer and then a cosine similarity layer is applied to

these two layers and then divided by the temperature value *self.temp* as described in the paper for SimCSE (Gao et al., 2021)

```
def predict_sentiment(self, input_ids, attention_mask):
    x = self.forward(input_ids, attention_mask)
    x = self.dropout(x)
    return self.classifier(x)


def predict_paraphrase(self,
                       input_ids_1, attention_mask_1,
                       input_ids_2, attention_mask_2):
    x_1 = self.forward(input_ids_1, attention_mask_1)
    x_1 = self.dropout(x_1)
    x_1 = self.paraphrase_1(x_1)
    x_2 = self.forward(input_ids_2, attention_mask_2)
    x_2 = self.dropout(x_2)
    x_2 = self.paraphrase_2(x_2)
    return self.cos_sim(x_1, x_2)/self.temp


def predict_similarity(self,
                       input_ids_1, attention_mask_1,
                       input_ids_2, attention_mask_2):
    x_1 = self.forward(input_ids_1, attention_mask_1)
    x_1 = self.dropout(x_1)
    x_1 = self.sts_1(x_1)
    x_2 = self.forward(input_ids_2, attention_mask_2)
    x_2 = self.dropout(x_2)
    x_2 = self.sts_2(x_2)
    return self.cos_sim(x_1, x_2)/self.temp
```

Further changes were also performed to optimize the training function mostly in the form of altering the loss function in order to fit the contrastive learning loss. For all three tasks, we utilized cross-entropy loss in order to fit the contrastive learning model. The equation for the loss function is given as:

$$\ell(x, y) = L = \{l_1, \ldots, l_N\}^\top, \quad l_n = -w_{y_n} \log \frac{\exp(x_{n,y_n})}{\sum_{c=1}^{C} \exp(x_{n,c})} \cdot 1\{y_n \neq \text{ignore\_index}\}$$

We can see that by using the cosine similarity of the two sentences as the input and using the truth labels as the target, we have achieved the loss function as described in the SimCSE study (Gao et al., 2021).

5 Experiments

5.1 Data

We will use the Quora and SemEval STS Benchmark datasets and their respective splits given in the CS 224N Default Final Project handout (CS224N, 2024). For the Quora dataset, this consists 400,000 string pairs and labels for whether they are paraphrases of each other – 141,506 train examples, 20,214 dev examples, and 40,431 test examples. For the SemEval STS Benchmark dataset, this consists of 8,628 string pairs of varying levels of similarity and its scoring from 0 to 5 denoting their similarity (0 denoting no relation, and 5 denoting an equivalent meaning) – 6,041 train examples, 864 dev examples, and 1,726 test examples.

5.2 Evaluation method

When testing the model, we will utilize the Pearson correlation coefficient between the true values and predicted values in order to assess its performance as was used in original SemEval paper. Since our goal was to improve our original minBERT model, we compared our results of the Pearson correlation coefficient of our new contrastive learning model to our old model, rather than make any comparisons to the results outlined by the SimCSE study (CS224N, 2024). To prove our work successful, we would hope to see a notable increase in the scores for all three downstream tasks from our minBERT model.

5.3 Experimental details

With the exception of using a GPU for training, all of the training parameters have been left as default from the provided code for **multitask_classifier.py**.

Variable Name	Default Value
sst_train	data/ids-sst-train.csv
sst_dev	data/ids-sst-dev.csv
sst_test	data/ids-sst-test-student.csv
para_train	data/quora-train.csv
para_dev	data/quora-dev.csv
para_test	data/quora-test-student.csv
sts_train	data/sts-train.csv
sts_dev	data/sts-dev.csv
sts_test	data/sts-test-student.csv
seed	11711
epochs	10
sst_dev_out	predictions/sst-dev-output.csv
sst_test_out	predictions/sst-test-output.csv
para_dev_out	predictions/para-dev-output.csv
para_test_out	predictions/para-test-output.csv
sts_dev_out	predictions/sts-dev-output.csv
sts_test_out	predictions/sts-test-output.csv
batch_size	8
hidden_dropout_prob	0.3

The model was pretrained with a learning rate of 1e-3 and finetuned with a learning rate of 1e-5. We also use a hidden size of 768. Additionally, in the research about SimCSE, it is determined that a temperature value of $\tau = 0.05$ yielded the most accurate results, and so that was the value that was used in training our model. The model was trained locally on an NVIDIA RTX 3070Ti FE using CUDA 11.8.

5.4 Results

The scores attained by both minBERT and the Contrastive Learning Method BERT model are shown below for the DEV set are shown below. Both scores are recorded after pretraining and finetuning using the parameters given in the **Experiment details** section above.

Downstream Task	minBERT	Contrastive Learning Method	Δ Accuracy
SST	0.520	0.532	+0.012
Paraphrase	0.550	0.805	+0.255
STS	0.053	0.529	+0.477

Below is the performance of the Contrastive Learning Method BERT model on the TEST set after pretraining and finetuning. The same parameters were used as described above.

Downstream Task	Contrastive Learning Method Accuracy
SST	0.524
Paraphrase	0.807
STS	0.495
Overall Score	0.693

As expected, we see substantial increases in performance in paraphrase detection and semantic textual similarity, as this was largely where changes in the neural network architecture were made. Specifically, we see the greatest improvement in semantic textual similarity predictions, and we believe that this occurs due to the weighting of the sentence pairs. While positive and negative pairs were given for paraphrase detection, we believe that the weighting of the sentence pairs for semantic textual similiarity yielded in greater changes when similar feature embeddings are pulled together and greater changes when dissimilar feature embeddings are pushed apart. Thus, we believe that this gives the model greater nuance when determining the magnitude to which a pair of sentences are related.

Surprisingly, we also see an increase in accuracy for sentiment analysis. We believe that this occurs due to the fact that the model attributes semantically similar sentences to have a similar sentiment score, leading to a slight bump in performance for this downstream task. The increase in performance for this specific task shows that the ability for contrastive learning to generalize input data for other potential downstream tasks as well when it comes to natural language processing.

6 Analysis

We believe that the large increase in accuracy when contrastive learning was implemented was due to the fact that the labels were already provided for us, and we did not need to generate our own positive and negative pairs. If we were to generate our own dataset, it might disagree with the pre-existing datasets regarding whether we believe two sentences are paraphrases of each other or how similarly we believe two senteces are to each other. Furthermore, if we were to arbitrarily create our own positive and negative pairs, we can accidentally change the coherence of a sentence or alter its meaning outright. For instance, if we were to have a sentence containing a negating word such as "not," if by change we were to remove this through random sentence augmentation, we would create a new sentence with the complete opposite meaning.

This project speaks largely to the ability of contrastive learning to generalize representations of data and proves its effectiveness more so than its ability to be trained on a smaller set of data. Since we used the same datasets for both the minBERT model and our new contrastive learning BERT model, we were unable to test if a smaller set of data can provide a similar accuracy when contrastive learning is implemented. However, because of the large increase in performance, it can be deduced that our new contrastive learning model can attain similar performances to the old minBERT model with significantly less training data.

While we had hope to implement certain input data augmentation techniques such as randomly removing words from a sentence or randomly replacing words with their synonyms, these techniques proved largely unfruitful and consistently resulted in significantly poorer accuracy. As mentioned in the SimCSE study, standard dropout yielded the most accurate result and we found this to be true in our own study, as well. Thus, there naturally exist limitations in the creation and delegation of positive and negative pairs. This is notably true for unsupervised learning.

7 Conclusion

Our findings demonstrate a significant increase in the accuracy of the model in all three downstream tasks. Notably, we see the greatest increases in accuracy semantic textual similarity and a substantial increase in paraphrase detection, which we expect from our change in architecture to the **predict_paraphrase()** function and the **predict_similarity()** function. The changes as described

above in the **Methods** section were made to give the model a better and more nuanced understanding of the textual relationship between the input data. Interestingly, these architectural changes yielded benefits beyond its intended scope and increased our accuracy for sentiment analysis, as well. However, it should be noted that the increase in score for sentiment analysis is not as substantial as the other two tasks, and can perhaps be worse with a different seed.

Due to the limited time for this project, we were unable to test further data augmentation techniques, but we have hope that such techniques exist that will yield better results than standard dropout, and we plan to continue further explorations in this avenue in future projects. Moving forward, we also hope to perform more testings to see whether other various loss functions can produce even better results when coupled with contrastive learning.

References

CS224N. 2024. Cs 224n default final project: minbert and downstream tasks.

Tianyu Gao, Xincheng 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*, page 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.