Sentence Similarity on Quora Question Pairs

Ayça Meriç Çelik - *21627103*, Okan Alan - *21526638*

Abstract—Semantic similarity is of the most popular NLP topics of recent years. After the great success of word embeddings and their applications, we made great progress in understanding natural languages. However, the challenge of evaluating the relations between sentences remains. There are a lot of promising approaches to this problem, and we aim to investigate them. In this paper, we will examine the task of "sentence pair classification" on Quora Question Pairs dataset with transformer-based models BERT and ALBERT.

1 Introduction

For our experiments, we decided to use the "Quora Question Pairs" (QQP) dataset, and we focused on the subproblem of semantic similarity named "Semantic Question Matching"[1].

Quora is one of the most popular question-answering platforms of our days. So it is not surprising that we observe many duplicate questions on that website. Different users ask the same question in different ways, but the overall intent and the meaning of the entry are the same. Thus, the expected answers would be the same. Detecting these duplicate questions can help both the users and the platform in many ways:

- The users could be redirected to the previously asked questions to reduce the waiting time for the answers.
- The platform could eliminate duplicate questions for the sake of compactness and simplicity.
- Various answers from different users would not be scattered to different entries.

This task is also a good and challenging example of semantic similarity measurement problem. Researchers have proposed different results from various approaches, such as Word2Vec [5], LSTM [6], etc. In our experiments, we fine-tuned transformer-based models for our pair classification task. We also examined the similarities between question vectors by using cosine similarity. In the end, transformer-based models produced pretty promising results.

2 Related Works

2.1 Word2Vec

When we look at sentence similarity history, word2vec [5] is the milestone for this NLP task. All models that are created after word2vec took as the base it. It takes sentences and then processes them to convert a matrix to achieve a word embeddings. In our task, we are comparing the similarity of sentences. It helps our base models that are BERT and ALBERT. Also, It is still a popular algorithm because of easy use.

2.2 LSTM

Long short term memory (LSTM [6]) was designed to avoid key limitations of RNN. It was used for a long time to examine the similarity of sentences. LSTM used in many NLP tasks. According to the works of Elior[x1], LSTM can take better than 80% accuracy in our dataset that is the quora question pair.

2.3 BERT

Bidirectional Encoder Representations from Transformers(BERT) [7] was published a few years ago and our accuracy on NLP tasks was level up via BERT especially it performs excellently on our problem that is sentence similarity and classification. We used the base uncased model of BERT and took it higher than 90% accuracy on the development/validation set.

2.4 ALBERT

A Lite BERT(ALBERT) [8] was developed to increase the speed of BERT and decrease memory usage. We used also ALBERT in our project and we observed the increasing speed in our own project. We got almost the same accuracy with BERT but ALBERT was a bit worse than BERT on our dataset.

3 Dataset

For our experiments, we decided to use the "Quora Question Pairs" (QQP) dataset, and we focused on the subtask of semantic similarity task, named "Semantic Question Matching" [1].

Quora is one of the most popular question-answering platforms of our days. So it is not surprising that we observe many duplicate questions on that website. Different users ask the same question in different ways, but the overall intent and the meaning of the entry are the same. Thus, the expected answers would be the same. Detecting these duplicate questions can help both the users and the platform in many ways: The users could be redirected to the previously asked questions to reduce the waiting time for the answers. The platform could eliminate duplicate questions for the sake of compactness and simplicity. Various answers from different users would not be scattered to different entries.

In 2017, Quora released the dataset with more than 400,000 pairs of questions. They also started a competition, and it attracted a lot of attention. Researchers, NLP enthusiasts, and students had a great opportunity to work with

organic data on semantic similarity tasks and compare the results with others easily. The dataset is still widely used and included in NLP benchmarks, such as QLUE.

For our research, we chose the GLUE benchmark [2] version of the dataset. It consists of three .tsv files as "train", "dev", and "test". There are 363192, 40372, and 390965 rows in the files respectively. Train and dev sets have 5 columns representing the first question's id, second question's id, the full text of the first question, full text of the second question, and the label indicating whether the questions are duplicate or not. The test set only contains two columns representing the full texts of first and second questions.

The best accuracy and F1 score obtained in GLUE benchmark are 86.5 and 66.1. [2]

4 Approach

We decided to use BERT-based transformer models for this task since we believe that these would produce good baseline scores for us.

BERT (Bidirectional Encoder Representations from Transformers) made a huge impact on NLP history. Compared to previous models, we can produce state-of-the-art results in various sentence-level and token-level tasks by just fine-tuning the model. It enables pre-trained deep bidirectional representations by using masked language models, so it is quite robust. These are all due to the model's architecture, which is a multi-layer bidirectional Transformer encoder. [3]

A Lite BERT(ALBERT) [8] was developed to increase the speed of BERT and decrease memory usage. Even though ALBERT-base model has only 12M parameters, an 89% less than BERT-base model, it is as robust as BERT!

We are able to use BERT/ALBERT in tasks like question answering and sentence pair classification since the model is able to take pairs of sentences as input. Our aim is to decide whether the two pairs of questions have the same intent or not, so we decided to use BERT/ALBERT as a sentence-pair classifier for our task.

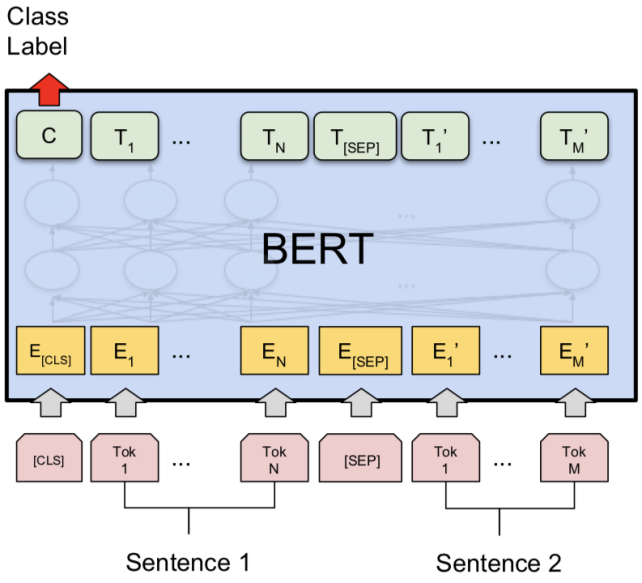

Fig. 1: Sentence Pair Classification tasks in BERT

5 Experiment

5.1 Preprocessing The Data

Question 1	Question 2	Duplicate?
How is air traffic controlled?	How do you become an air traffic controller?	No
Why doesn't everyone I ask to answer a question-answer it on Quora?	Why doesn't anyone here answer my questions?	Yes

TABLE 1: The raw examples from the dataset are as follows

We followed the steps below to prepare our dataset:

1) Dropped the faulty lines (3 lines in total).
2) Dropped "qid1" and "qid2" columns in the train and dev set.
3) Reset the indexes in three sets.
4) Converted all texts to lowercase. (We used uncased models for the training).
5) Expanded the English contractions: As an example, "couldn't" becomes "could not". This is quite important since we later insert spaces before and after the punctuation marks. With this, we were able to protect the contractions from deformation.
6) Punctuation isolation: We insert a space before and after each punctuation. We prefer to keep the punctuation marks since BERT performs better with them. Instead, we separated them from words to be correctly tokenized by BERT Table2.

Question 1	Question 2	Duplicate?
how is air traffic controlled ?	how do you become an air traffic controller ?	No
why does not everyone i ask to answer a question answer it on quora ?	why does not anyone here answer my questions ?	Yes

TABLE 2: The examples after preprocessing

5.2 Approach #1 - Thresholding The Similarity Scores of Vectors

In order to review the quality of BERT sentence vectors, we encode text pairs without any fine-tuning. Later, we computed the cosine similarity of these two sentence vectors and compared them with the ground-truth label. We used Pytorch implementation of BERT from HuggingFace [4] for this task.

Since getting the sentence embedding is relatively slow, and our main goal is to train a classifier, we did not spend so much time on this part. We only encoded the dev set and calculated the accuracy by defining a threshold value for similarities by hand. Table3

The threshold of 0.85/0.90 gave the best result. However, there are some problems. Let us review these false positive examples Table4:

Threshold	Accuracy
0.70	0.624
0.75	0.668
0.80	0.704
0.85	0.728
0.90	0.728
0.95	0.697

TABLE 3: threshold - accuracy

Question 1	Question 2	Similarity	Duplicate?
why is my life getting so complicated ?	why is my life so complicated ?	0.989	No
why are african - americans so beautiful ?	why are hispanics so beautiful ?	0.835	No

TABLE 4: False Positive Examples

Although some question pairs have very high similarity scores, it does not necessarily mean that these two sentences are duplicate. The questions can have similarities in terms of meaning, however the intent of the question may differ. So, the answers of these questions may differ. This means that these two questions are not duplicates, but this method cannot differentiate that, and produce false-positive resultsTable5.

Question 1	Question 2	Similarity	Duplicate?
what are your views about demonetisation in india ?	what do you think about the ban on 500 and 1000 denomination notes in india ?	0.595	Yes
how can i get rid of a bad habbit ?	what are good strategies for getting rid of a bad habit ?	0.642	Yes

TABLE 5: False Negative Examples

Although similarity scores are generally high and the model tends to produce false positives, it can produce false negatives as well. Sometimes, the model cannot capture the desired relation between questions.

As we can see, this method is not powerful enough to classify the pairs correctly. Additionally, thresholding the outputs manually is not the optimal way to handle the predictions. We need to automate this task by training a classifier.

5.3 Approach #2: Fine-Tuning BERT & ALBERT

To train a sentence-pair classifier, we fine-tuned BERT and ALBERT on our train set, and calculated accuracy on the dev set. We also got the test set prediction for the future evaluation on the GLUE benchmark. We used "SimpleTransformers"[5] module, which is a high-level implementation built on HuggerFace Transformers, for the sake of simplicity and efficiency. We trained our model in the free servers of Google Colab (Tesla P100).

We got our best results with the batch size as 16 and epoch count as 3. We experimented with different parameters, and the results as follows:

- **Learning Rate**: Default $4e^{-5}$. Increasing the learning rate (as $4e^{-4}$) decreased the accuracy significantly, so we left it as the default value.
- **Epoch:** We trained our model in 3 and 2 epochs, and as we expected, higher epoch gave higher accuracy.
- **Batch Size:** Since models are quite large, the highest batch size that we can use was 16. We were able to use batch size of 32 during the training of "albert-base-v2" only.
- **Models:** We experimented with "bert-base-uncased", "bert-large-uncased", "albert-base-v2" and "albert-large-v2". The larger models such as "albert-xlarge-v2" and "albert-xxlarge-v2" either did not fit into memory, or did not produce good results. Our guess on the reason that the bigger models performs poorer is that it is unnecessarily complex to be trained on our dataset.

5.3.1 *Experiment #1: Only 1000 Training Examples*

Epoch	Method	Accuracy	Eval Loss
3	albert-base-v2	0.798	0.509
2	albert-base-v2	0.785	0.444
3	bert-base-uncased	0.765	0.503
2	bert-base-uncased	0.740	0.511

TABLE 6: Cosine Similarity & Thresholding results.

As you can see, it outperformed our previous thresholding method with only 1000 training examples and 2 epochs. It convinced us that we are on the right track. We also observed that in the 3rd epoch, the model continues to train. In our further experiments, we aim to increase the epoch number.

5.3.2 *Experiment #2: Different Model Versions*

Since the training takes a quite long time, we experimented with different BERT/ALBERT versions with 10.000 training examples at maximum. We trained "albert-xlarge-base" and "bert-large-uncased" in this stage.

Epoch	Method	Accuracy	Eval Loss
3	albert-large-v2	0.631	0.650
3	bert-large-uncased	0.631	0.663

TABLE 7: Training results with only 1000 examples.

As you can see, since the models are larger, it is unable to converge with few training examples. Let us examine all the models trained with 10.000 examplesTable8:

Epoch	Method	Accuracy	Eval Loss
3	bert-base-uncased	0.837	0.650
3	bert-large-uncased	0.836	0.504
3	albert-base-v2	0.830	0.427
3	albert-large-v2	0.631	0.658

TABLE 8: Training results with only 10.000 examples.

It seems like larger models did not improve the accuracy scores. For the sake of efficiency, we continued to experiment with the base models.

5.3.3 *Experiment #3: Full Training Set*

The training of each model took 5 hours. The results are as follows:

Epoch	Method	Accuracy	Eval Loss
3	albert-base-v2	0.908	0.278
3	bert-base-uncased	**0.911**	0.344

TABLE 9: Full-set training results.

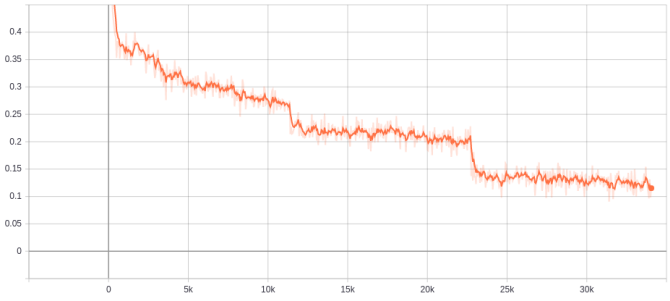

Fig. 2: ALBERT's Loss Graph

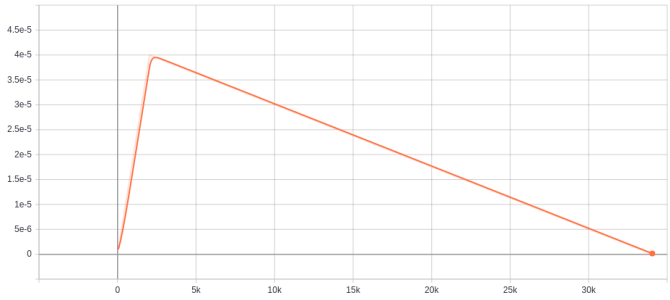

Fig. 3: ALBERT's Learning Rate Graph

We reached an accuracy score of **%91**! As you can see, more training example yield a more powerful classification

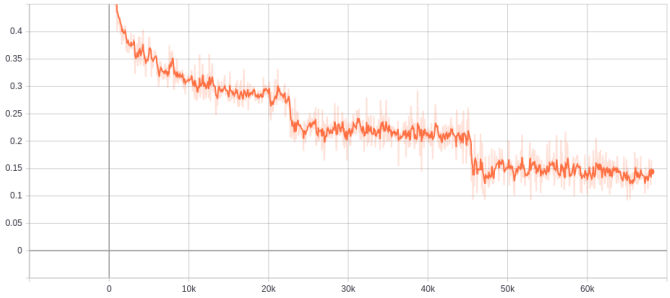

Fig. 4: BERT's Loss Graph

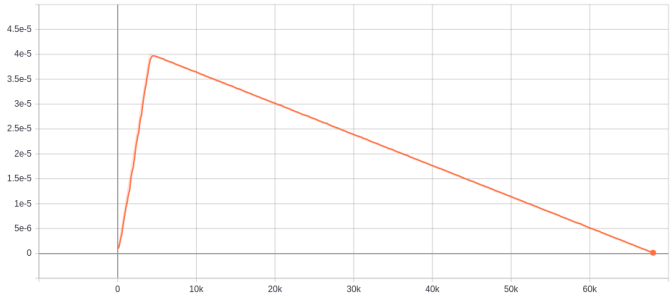

Fig. 5: BERT's Learning Rate Graph

model. Since our time was limited due to Google Colab's restrictions, we were not able to train our model with higher epochs, but we are expecting higher accuracies with more epochs as well!

6 Conclusion

We observed that transformer-based models such as BERT and ALBERT perform quite well in the task of sentence pair classification. By just fine-tuned the model's classifier on our dataset, we were able to get the accuracy of %91. Both BERT and ALBERT produced better results on fewer examples than the thresholding method we tried earlier. This proves two things: fine-tuning really makes difference, and deep neural networks outperforms simple manual approaches -as we expected-. We also saw that the larger versions of BERT and ALBERT are unnecessary to use in our dataset since they increase the training time and resource need quite a lot without improving the scores much. In the future, we will experiment with higher epochs and different models as well, and we are hoping that we will outperform our current results.

References

[1] Nikhil Dandekar: February 13, 2017 *Semantic Question Matching with Deep Learning*

[2] Alex Wang1, Amanpreet Singh1, Julian Michael2, Felix Hill3, Omer Levy, Samuel R. Bowman *GLUE: A MULTI-TASK BENCHMARK AND ANALYSIS PLATFORM FOR NATURAL LANGUAGE UNDERSTANDING*

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, Illia Polosukhin *Attention Is All You Need*

[4] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Jamie Brew *HuggingFace's Transformers: State-of-the-art Natural Language Processing*

[5] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean *Efficient Estimation of Word Representations in Vector Space*
[6] Felix A. Gers, Jurgen Schmidhuber, Fred Cummins *Learning to Forget: Continual Prediction with LSTM*
[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*
[8] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, Radu Soricut *ALBERT: A LITE BERT FOR SELF-SUPERVISED LEARNING OF LANGUAGE REPRESENTATIONS*