Fine-Tuned Sentence Transformer Model for Question Answering Task

Ercong Nie
Center for Language and Information Processing,
University of Munich (LMU)
e.nie@campus.lmu.de

Abstract

Nowadays, various pre-trained language models based on representation learning are constantly being developed and have achieved splendid performance in the application of various natural language processing (NLP) tasks by fine-tuning. Sentence Transformer models provide with the possibility to embed sentences and compare the semantic similarity between sentences by creating a siamese and triplet networks based on pre-trained Transformer models. The paper fine-tunes a Sentence Transformer model and applies it to the question answering (QA) task by answer selection. To be more specific, we use the fine-tuned sentence transformer model to select the proper answers to a given question from a given pool of candidate answers. The experiment results show that by fine-tuning, the accuracy can be improved significantly from 0.2664 to 0.4867. Further research should be done on fine-tuning more models and training on more domain data.

1 Introduction

Language representation models have been a popular topic and constantly show its power in various fields of natural language processing (NLP) tasks with the rapid development of deep neural networks in the past years. The network architecture Transformer, which is based solely on attention mechanisms and entirely dispenses recurrences and convolutions (Vaswani et al., 2017), is commonly applied by many language models. Language models are usually pre-trained language representations from a large amount of unlabeled texts, such as BERT (Devlin et al., 2019) and many of its variants like RoBERTa (Liu et al., 2019), TinyBERT (Jiao et al., 2020) etc. By fine-tuning the pre-trained models, many downstream tasks can be processed, such as text classification, sequence labeling, language inference, machine translation and so on.

Most language models operate on the word level, i.e. represent words with a series of numbers, namely vectors. A common method used to represent words and their semantic meanings and syntactic information in a context is the word embedding. However, language representation can be carried out on a higher level, that is to say, sentences and even documents can also be represented or embedded, corresponding to sentence embedding and document embedding. Sentence Transformer is a model based on the Transformer for state-of-the-art sentence embeddings, which can be for certain tasks including large-scale semantic similarity comparison, clustering and information retrieval via semantic search (Reimers and Gurevych, 2019).

Question answering (QA) is a classical NLP task and can be applied in many fields, such as search engine, chatbot, information retrieval, conversational AI etc. QA is basically a system that allows a user to ask a question in natural language and receive an answer quickly and succinctly, with sufficient context to validate the answer (HIRSCHMAN and GAIZAUSKAS, 2001). Currently, most influential spoken dialog system include a question answering module, e.g. Apple's Siri, IBM's Watson and son on. Feng et al. (2015) previously looked at QA from an answer matching and selection perspective. This paper adopts their perspective towards QA as well.

This paper combines the Sentence Transformer and QA. We apply Sentence Transformer model to the QA task by sentence matching and semantic similarity comparison. The goal is to train a sentence transformer model which can be used to select the proper answers to a given question from a given pool of candidate answers. The goal can then be divided into two subtasks: (1) How is the sentence transformer model trained? (2) How is the model applied to answer selection? We firstly select a pre-trained Sentence Transformer model and then fine-tune it with the training QA dataset. After comparing several different language models, we select the TinyBERT and base our Sentence Transformer model on it. Each question in the training dataset is aligned with a true answer labeling 1 and a false answer labeling 0. In this way, a training dataset composed of sentence pairs and labels is constructed. The model is fine-tuned by maximizing the cosine similarity between question and true answer and minimizing the cosine similarity between question and false answer. The cosine similarity loss function is adopted. With the fine-tuned Sentence Transform model, we evaluate its performance on test QA dataset. In the test QA dataset, each question is provided with a pool of candidate answers and a set of truth answers. We calculate the similarity between the question and every candidate in the pool using the fine-tuned Sentence Transformer model and then select the best candidate as true answer. If the selected answer lies in the set of truth answers, then the question is correctly answered by the QA system, otherwise wrongly. Last but not least, we calculate the accuracy of the overall test data.

The paper is structured in the following way: Section 1 introduces the background and basic information of the work. Section 2 is about the previous work related to the topics in this paper. Section 3 explains the specific methodology of the process of the work and how the model is established. In section 4, we evaluate the performance results of the methodology on QA tasks. Section 5 concludes the work in this paper.

2 Related Work

In this section, we first introduce the study on language model, which is the basis of the work. Then, we discuss the study on sentence embedding methods. The last part is on the study of QA tasks.

2.1 Language Representation and Language Models

The state-of-the-art pre-trained language models can well represent language and significantly improve the performance of various tasks in the NLP field. BERT is a pre-trained deep bidrectional Transformer network using two unsupervised pretraining tasks, Masked LM and Next Sentence Prediction (Devlin et al., 2019), so that a model representing the internal syntactic and semantic information of language can be created.

Many further studies focus on the improvement of BERT and some variants of BERT performing better than the original base in specified fields of tasks are given to birth. RoBERTa (Liu et al., 2019) optimized BERT by measuring the impact of hyperparameters and training data size. Jiao et al. (2020) introduced the knowledge distillation (KD) method of Transformer-based models into TinyBERT, which transferred large BERT model to a smaller one.

BERT network structures are based on word embeddings. No independent sentence embeddings are computed. BERT has its setup dealing with the similarity of sentence pair by inputting BERT two sentences separated by a special token and can be applied on the Semantic Textual Similarity (STS) benchmark (Cer et al., 2017).

2.2 Sentence Embeddings

A simple method to produce sentence embeddings is averaging the word embeddings of words in a sentence into a sentence embedding, for example, Joulin et al. (2016) set a baseline using the method for text classification. However, this method is not powerful enough when dealing with more complicated tasks.

Researchers tried several different kinds of methods to derive sentence embeddings from the outputs of language models. May et al. (2019) passed single sentences through BERT and derived a fixed sized vector by averaging the outputs. Qiao et al. (2019) used the output of a special token as the sentence embedding to represent the sentence in their study. However, according to Reimers and Gurevych (2019), there is not evaluation or evidence yet on how useful are the sentence embeddings created in these methods.

Akbik et al. (2019) developed an NLP framework, FLAIR, designed to simply mix and match different types of word embeddings with minimal effort, where two types of sentence embeddings are also included, i.e. Document Pool Embeddings and Document RNN embeddings. Document Pool Embeddings do a pooling operation over all word embeddings in a sentence to obtain an embedding for the whole sentence. Document RNN Embeddings run a recurrent neural network (RNN) over all words in sentence and use the final state of the RNN as embedding for the whole document.

Reimers and Gurevych (2019) presented SBERT model to yield useful sentence embeddings by fine-tuning pre-trained BERT network. SBERT firstly adds a pooling operation to the output of BERT to derive a fixed sized sentence embedding. Then, the weights and parameters are updated by creating a siamese and triplet networks (Schroff et al., 2015). After the fine-tuning, the produced sentence embeddings are more semantically meaningful, can be compared with cosine similarity and applied to specific tasks like QA.

2.3 Question Answering

In information retrieval and NLP, QA is the task of automaticcaly providing an answer for a question asked by a human in natural language (Bouziane et al., 2015). Lopez et al. (2011) divided QA task into three subtasks: question analysis, document retrieval and answer extraction.

Ishwari et al. (2019) reviewed the development of natural language question answering. Traditional question answering systems were logical representations of decision trees based on grammatical rules, reflecting the way humans understand text. This approach is called rule-based approach. Ontology based QA systems take queries expressed in natural language and a given ontology as input, and return answers drawn from one or more knowledge bases that subscribe to the ontology. Deep learning methods allow a machine to be fed with raw data and to automatically discover the representations needed. Induction of neural networks for QA brings more possibilities. Feng et al. (2015) applied CNN-based system to address the non-factoid question answering task in the insurance domain.

3 Methodology

The paper fine-tunes a pre-trained Sentence Transformer model for answer selection, a question answering task. To be more specific, the goal of the work is to train a sentence transformer model which can be used to select the proper answers to a given question from a given pool of candidate answers. The goal can then be divided into two subtasks: (1) How is the sentence transformer model trained? (2) How is the model applied to answer selection? The original train and evaluation data set have the same format, but are used and processed differently in two subtasks. More datails on the format of the data set are introduced in the following subsection 3.1

The sentence transformer model is built based on a pre-trained transformer model. The structure of a sentence transformer model is shown in Figure 1. The siamese networks of a sentence transformer model are created based on pre-trained language models. We input two language models with the same parameters separately sentence A and B. After pooling layer, two sentence embeddings u and v are created and then the cosine similarity between u and v is computed. The objective function used in the model is mean-squared-error (MSE) loss. The fine-tuned Sentence Transformer model can then be applied for insurance QA task by calculating the sentence pair similarity between question and candidate answers in the pool. The returned answer by the QA system is the candidate with the highest cosine similarity.

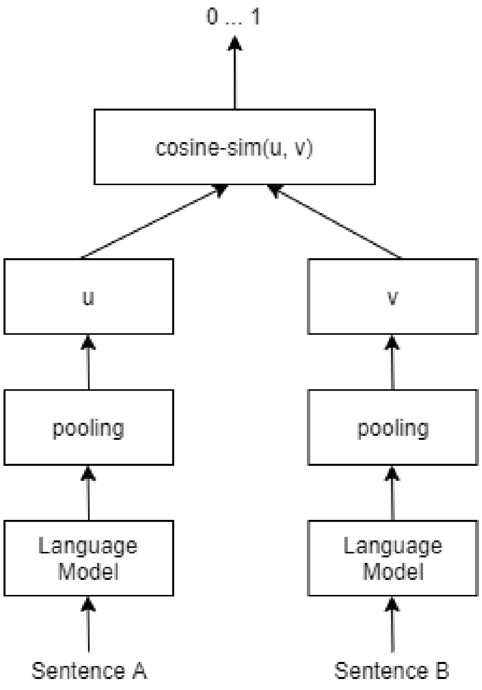

Figure 1: Sentence Transformer Architecture.

3.1 Dataset Description

The dataset used in the work is InsuranceQA Corpus[1], which was created by researchers of IBM Watson who applied it to training a CNN model to solve QA task (Feng et al., 2015). It contains questions and answers in the insurance domain from the website Insurance Library[2]. The contents of the corpus are the user questions from the real world and the answers with high quality composed by professionals with expert domain knowledge. In the data set, each question has a pool of 100 possi-

[1] https://github.com/shuzi/insuranceQA

[2] https://www.insurancelibrary.com/

No.	Model Name	Accuracy	Running Time
1	average_word_embeddings_komninos	0.0875	**58.271 s**
2	paraphrase-distilroberta-base-v2	0.2417	4440.939 s
3	paraphrase-MiniLM-L3-v2	0.2241	1182.681 s
4	paraphrase-MiniLM-L6-v2	0.2447	1827.053 s
5	paraphrase-MiniLM-L12-v2	0.2661	3117.037 s
6	paraphrase-mpnet-base-v2	**0.2825**	9489.678 s
7	paraphrase-TinyBERT-L6-v2	0.2664	5069.734 s
8	stsb-mpnet-base-v2	0.1575	7183.301 s
9	stsb-roberta-base-v2	0.1507	6456.512 s

Table 1: Performance of different pre-trained Sentence Transformer models on the insurance QA task.

ble answers, some of which are true answers (the so-called ground truth). The pool is built by using a general search engine like Google Search or an information retrieval software library like Apache Lucene and the data set is created by collecting question and answer pairs from the internet.

The corpus is composed of training and evaluation two parts. In the original dataset, the tokens are represented by their indexes instead of character forms. For all tokens starting with idx_, we need to refer to the vocabulary file offered by the corpus for the corresponding word. The training and evaluation files have the same format: <Domain><TAB><QUESTION><TAB> <Groundtruth><TAB><Pool>. <Domain> means the subclass of the question topic. <question> is represented by the tokens in the form of index. <Groundtruth> is the set of correct answers and <Pool> consists of all possible answers. Both <Goundtruth> and <Pool> is represented by the label of answers, so there is an extra file mapping labels to answers (represented by token indexes) in the corpus.

There are 12,889 questions with 107,889 running words in the training set and 4,000 questions with 33,746 running words in the evaluation data. Each question has a pool of answers with the number of 100. There are totally 27,413 answers in the answer set file with 3,065,492 running words.

3.2 Data Preparation

The data preparation work is divided into two parts. The first part is to create training data from the corpus for the model fine-tuning. The second is to prepare the evaluation data for the evaluation of the QA system.

Preparation of Training Data To train the sentence transformer model, sentence pairs with label are required. So for each question in the original train data set, we create two sentence pairs to compose the train data set that can be directly applied to training the sentence transformer model. One sentence pair has a positive label, i.e. 1. We selected the first answer from the <Goundtruth> (the correct answer set) to compose the positive sentence pair. On the contrary, the other sentence pair has a negative label, i.e. 0. We randomly selected an answer from the wrong answers to compose the negative sentence pair.

Preparation of Evaluation Data Evaluation data are used to evaluate the accuracy of the QA task of the fine-tuned model. What we do here is transfer index sentences into word sentences and match question label to its ground truth and pool.

3.3 Model Selection

Regarding fine-tuning the model, the first step we do is select a pre-trained Sentence Transformer model for later fine-tuning on the insurance QA dataset. Tabel 1 shows the evaluation result of different pre-trained Sentence Transformer models on the insurance QA task.

In Model 1, the sentence embeddings derived from word embeddings simply by mean pooling are used. Since it does not use the neural networks, the running time is very fast, while the accuracy is rather low. Model 2-7 use the same training dataset, which has a variety of sources, such as AllNLI, SimpleWiki, Yahoo answers title question etc. Model 8-9 use the benchmark of NLI and STSb as training data. Model 3-5 are fine-tuned based on the MiniLM language model of Mircosoft. Model 2, 7 and 9 use the variant of BERT as base model. Model 6 and 8 use the MPNet model of Microsoft, which combines the advantage of BERT and XLNet by unifying the trained tasks of both

Pre-trained Model without Fine-Tuning	
Name	paraphrase-TinyBERT-L6-v2
Accuracy.	0.2664
Runtime	5069.734 s
Pre-trained Model with Fine-Tuning	
Name	paraphrase-TinyBERT-L6-v2
Accuracy.	0.4867
Runtime	5651.248 s

Table 2: Result of fine-tuned Sentence Transformed applied to the insurance QA task.

models, masked language modeling and permuted language modeling, in one view (Song et al., 2020).

According to the result in Table 1, Sentence Transformers trained on the paraphrase have overall better performance than models trained on NLI and STS. The reason could lie in that paraphrase training tasks are more similar to the QA tasks. Both are related to sentence pair matching. Taking both accuracy and running time into account, we finally select model 7 “paraphrase-TinyBERT-L6-v2” as the pre-trained Sentence Transformer model to be fine-tuned for the insurance QA task.

3.4 Fine-Tuning Details

We fine-tune the pre-trained Sentence Transformer model “paraphrase-TinyBERT-L6-v2” using the training data introduced in section 3.2. The training dataset is a collection of 25,778 question answer pairs, half of which are annotated with label 1, the other half with label 0. We used a batch size of 64, one epoch and cosine similarity loss and a linear learning rate warm-up over 10% of the training data. We also set an evaluator composed from 1,000 sentence pairs in the training data to evaluate the model during the training. The number of evaluation steps is 2,000.

4 Evaluation

The role of the well-trained (fine-tuned) sentence transformer model in the QA task (i.e. answer selection from the pool) is to compute the sentence similarity. At the beginning, we tried several methods of applying the fine-tuned Sentence Transformer model to address the QA task. Since the answers in the data set are often quite long and could even be composed of several sentences, we considered if it would be better to sentence-tokenize the answer firstly and then compute the similarity between question and each single sentence in the answer. However, after several trials, we found the accuracy was significantly lower than regarding the whole answer as one ”big” sentence.

We also attempted to collect all tokenized sentences and to rank the cosine similarity between all sentences and a question. Then we selected the n best sentences and calculated how many selected sentences each answer contains. At last, we select the answer with the most sentences to be the best answer. However, not only is the efficiency of this method less than the original one, but its accuracy is also worse.

4.1 Evaluation Method

The evaluation method we finally adopted is the most straightforward one. The fine-tuned Sentence Transformer is applied directly to the question and the candidate answers in the pool. Then, we select the answer with the highest cosine similarity and check if the answer is in the ground truth set, which contains all true answers. If the selected answer indeed belongs to the ground truth, then the question is supposed to be answered correctly, otherwise wrongly. At last, we count the accuracy of the evaluation dataset and use it as the the evaluation metrics.

4.2 Results

From Table 2 we can see that by fine-tuning with the domain data, the accuracy of the QA system has been significantly improved. It rises from 0.2664 to 0.4867 with an increasing of 82.7%. It proves that the Sentence Transformer model has learned a stronger language representation ability in the insurance domain during the fine-tuning process. It should also be noted that the experiment only performed on very few models and tried no more sets of hyperparameters. It is clear that with more attempts on hyperparameter selection, the model can achieve even better performance.

5 Conclusion

In this paper, we combined the Sentence Transformer and QA task. We fine-tuned a Sentence Transformer Model and applied it to the QA task. For that, we designed an evaluation method to introduce Sentence Transformer model by sentence matching and semantic similarity. When selecting the model to be fine-tuned,we compared the performance of different pre-trained Sentence Transformer models on solving QA tasks. We used the accuracy as metrics. The results show that the Sentence Transformer model can be better applied in the QA tasks by fine-tuning.

Limited to the computational resources, we failed to perform on more models and did a deeper research on different models. Besides, we did not study the effect of different hyperparameters on the performance. This is what can be done in the future work. Moreover, we could also directly fine-tune a Sentence Transformer model based on a basic pre-trained language model instead of fine-tuning an existing pre-trained Sentence Transformer mode. The comparison of the performance between the both is also worth a research.

References

Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 54–59, Minneapolis, Minnesota. Association for Computational Linguistics.

Abdelghani Bouziane, Djelloul Bouchiha, Noureddine Doumi, and Mimoun Malki. 2015. Question answering systems: survey and trends. *Procedia Computer Science*, 73:366–375.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Minwei Feng, Bing Xiang, Michael R. Glass, Lidan Wang, and Bowen Zhou. 2015. Applying deep learning to answer selection: A study and an open task.

L. HIRSCHMAN and R. GAIZAUSKAS. 2001. Natural language question answering: the view from here. *Natural Language Engineering*, 7(4):275–300.

K. S. D. Ishwari, A. K. R. R. Aneeze, S. Sudheesan, H. J. D. A. Karunaratne, A. Nugaliyadde, and Y. Mallawarrachchi. 2019. Advances in natural language question answering: A review.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tinybert: Distilling bert for natural language understanding.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Vanessa Lopez, Victoria Uren, Marta Sabou, and Enrico Motta. 2011. Is question answering fit for the semantic web?: a survey. *Semantic web*, 2(2):125–155.

Chandler May, Alex Wang, Shikha Bordia, Samuel R. Bowman, and Rachel Rudinger. 2019. On measuring social biases in sentence encoders.

Yifan Qiao, Chenyan Xiong, Zhenghao Liu, and Zhiyuan Liu. 2019. Understanding the behaviors of bert in ranking.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. *arXiv preprint arXiv:2004.09297*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.