Natural Language Processing Project Report

1 Introduction

There's no denying that climate change is a hot topic nowadays. However, the increasing unverified statements related to climate science are distorting public opinions while they are not rigorous enough to discuss this problem. As human beings with scientific thoughts, when we want to verify a claim, we will first find out the reliable related evidences, and then judge whether they support this claim or not. Now it is a time to make this laborious process completed automatically.

The system can be broken into two tasks: the first task is to find the relevant evidences given a claim; and the second task is to classify the relationships between the claim and the evidences.

For the first task, we implemented an evidence-fetching method which uses the pre-trained language model, BERT (Devlin et al., 2019), and one of the famous (Approximate Nearest Neighbour) ANN algorithms, FAISS, to find the relevant evidences efficiently.

For the second task, we built a classification model which was again fine-tuned from a language model based on BERT, adding one classification layer at the top of the BERT model.

Several other methods are also considered and experimented, with details explained in the following sections.

2 Task 1: Evidence Retrieval

We want to fetch as high accuracy as possible, but it is also worth noticing that our dataset is relatively large and as a user-friendly system (and the ability to train our model) it is importance to limit the waiting time in a proper duration. So for this task efficiency is also included in our analysis.

2.1 Method 1 - BERT embedding with ANN

A plain way to measure the relativeness of a claim and an evidence is to calculate their similarity. To do the calculation numerically, we first need to represent our sentences as vectors, and here comes the $Sentence\,Transformer$ model (Reimers and Gurevych, 2019), which is based on a siamese BERT Network. We have 1208827 different evidences, which is a large number if we want to compare each claim (1228 in total) with every evidence one by one. Therefore after calculating all the embeddings for both claims data and the evidences, we use $faiss$ library to search for the top 100 nearest (or most similar) evidences for each test claim. This will reduce our candidate evidences from 1000,000 to 100,000, which is faster for the last process. Finally, we iterate all the claims again and try to accurately find the most similar evidences from the 100 candidates using cosine similarity metric.

Since the efficiency is the most importance part in this method, we recorded the average running time of this model:

Process	Avg Time
Embedding (BERT)	322.495s
Build Index (FAISS)	4s
Search for ANN (FAISS)	94s

Table 1: Time Recording for Task 1 Method 1

2.2 Method 2 - Siamese Sentence Transformer

The first Method is good to provide us with a quick result of the first task. However, it did not perform well. Both approximation process and the untailorred pre-trained weights are the probable reasons for the failure.

To train a self-defined model to capture the preferred similarity information in our peticular goal, we adapted the Siamese Sentence BERT model (Reimers and Gurevych, 2019). Sentence-BERT (SBERT) is a modification of the pretrained BERT network that use siamese and triplet network structures to derive semantically meaningful sentence

embeddings that can be compared using cosine-similarity. Its architecture is as follow:

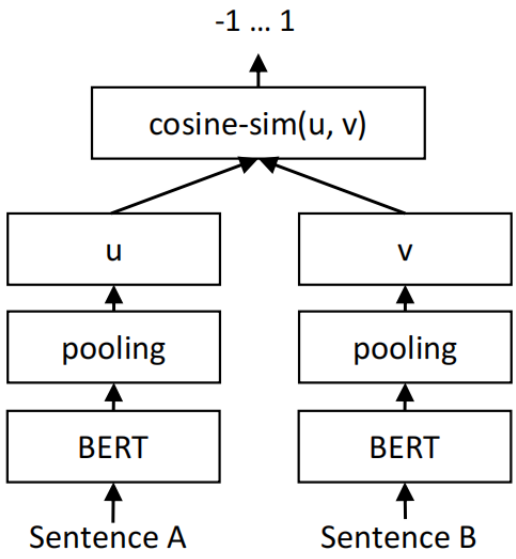

Figure 1: Sentence-BERT Architecture

The inputs are two sentences to be compared, after embedding and pooling them with 2 BERT models separately, we will combine them together and calculate the cosine similarity of them. The loss function for this problem is specifically designed by (Khosla et al., 2021) as follow:

http://yann.lecun.com/exdb/publis/pdf/hadsell-chopra-lecun-06.pdf

It aims to learn embeddings that push similar pairs of samples closer together and dissimilar pairs further apart and assigns a small loss to similar pairs and a large loss to dissimilar pairs. In the context of this task, the similar pairs are claim-evidence pairs that are relevant, and dissimilar pairs are claim-evidence pairs that are irrelevant.

2.3 Trick - Negative Sampling

Whta worths mentioned is that for better performance and shorter training time, we also implemented Negative Sampling here. That is, instead of using the whole negative training data (i.e., all claim-evidence pairs that are not correlated), here we only randomly sampled 20 negative samples from the uncorrelated evidences to train our model. It not only helps us not get stuck by the imbalanced data set (namely, the huge amount of the uncorrelated pairs), but also saves us a lot of time during training process.

2.4 Trick - Dropout

Deep learning neural networks are likely to overfit a training dataset with few examples quickly. Ensembles of neural networks with different model configurations are known to reduce overfitting, but require the additional computational expense of training and maintaining multiple models. A single model can be used to simulate having a large number of different network architectures by randomly dropping out nodes during training. This is called dropout and offers a very computationally cheap and remarkably effective regularization method to reduce overfitting and improve generalization error in deep neural networks of all kinds.

In our implementation, all of our models were set to a default dropout rate through the $Trainer$ function imported from $SentenceTransformer$ library.

3 Task 2: Claim Classification

3.1 Method 1 - BERT embedding with Classifier

It still makes sense to use BERT model to encode our sentences, since the model is really good at this, so the task needs to be done here is to modify the model a little to fit our down stream problem. We add a multiclassfication layer at the top of the BERT model, with $[[CLS]$, $claim$, $[SEP]$, $evidences$, $[SEP]]$ concatenated together as the inputs, and a vector of logits indicating the the probablity (before softmax) as the output.

3.2 Method 2 -Rule-based Classifier (not implemented)

From careful analysis we found that the claim labels are of some rule, for example, if all the evidences are for the claim, then the claim label is $SUPPORT$, and if all the evidences are against it, its label would be $RERUTES$. $DISTRUBE$ stands for some of the evidences support it and other are not, whilst NOT_ENOUGH_INFO are the kind of claims that are hard to decide their labels.

Based on these rules, it may be possible for us to firstly build a sentiment analysis classifier (whether the evidence shows positive or negative attitude to its corresponding topic), then automatically classify them as the aim labels. However due to the time limitation, this method is just a script and has not been implemented in this project.

4 Experiment Results

We have tried several types of methods and try to compare their performance. Here are some of our results. All of the results are tested on the development data set.

Method	THRESHOLD	TOP-K	F1
Method 1	0.81	5	0.057
Method 1	0.85	5	0.7757
Method 1	0.89	5	0.0849
Method 1	0.91	5	0.0817

Table 2: Experiment Results for Task 2 Method 1

Here the THRESHOLD refers to the threshold we used to consider whether an evidence is related or not, and the TOP-K stands for the max number of evidences that a claim can have.

EPOCHs	Accuracy
4	0.4415
20	0.5389
300	0.5519

Table 3: Experiment Results for Task 2 Method 1

Note that there are some other experiments that we have done, but none of them have performed well (or at least better than the one listed above). This may be due to a lack of experiments on the hyperparameters and better fine-tune the models' structure. Due to the time limitation, we haven't implemented too much models and the final evaluation results on Kaggle is 0.0625.

4.1 References

References

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

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2021. Supervised contrastive learning.

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