COMP90042 Web Search and Text Analysis
Automatic Fact Verification

806719 Yiming Qiu
869895 Mingchi Zhang

1 Introduction

Fact verification could be regarded as a natural language processing problem which helps people automatically identify the correctness of a claim sentence. This report explains an approach to do this with two stages: sentence retrieval and label inference.

Sentence Retrieval: For the given claims, retrieve sentences which have relevant meanings.

Label inference: Predict the label for each claim with their evidence. This is a 3 classes prediction. The labels are: *SUPPORTS*, *REFUTES*, *NOT ENOUGH INFO*.

In this report, the methods which are used in this system will briefly go through. After showing the result of the system performance, the error analysis will be discussed. In the end, this report will list several possible improvements.

2 Fact extraction and verification

This section will explain how the system works and the background of underlying approaches.

2.1 Sentence retrieval

In this stage, an information retrieval system is built for fact extraction. As the following figure shows, the system extracts the evidence sentences for certain claims. Therefore, this stage generates the input files of the next stage: label inference.

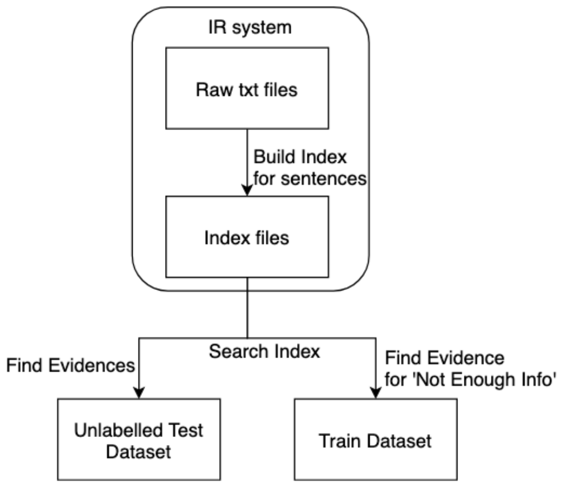

Figure 1: Sentence retrieval

2.1.1 Technique background

In this IR system, we use several tools to build the index and search the sentence.

PyLucene: It is a python extension which allows users to access java Lucene functions in python. We use this to build indexes for each sentence. PyLucene has the ability to process the sentence such as stemming, lowercase, stop words filtering.

spaCy: It is a python library which provides function of POS tagging, sentence dependency parsing and more. This IR system uses spaCy to extract the entity from claim sentences.

2.1.2 Sentence index

As figure 1 shows, this IR system firstly generates the index for each sentence. For each sentence, we build two indexes for it, sentence id and document name. The sentence id is defined as a composite element which includes document name and sentence identifier. The sentence id is used to locate a sentence content.

As to document name, we reckon it is the most important information for searching. The document name will be used for searching evidence sentences.

2.1.3 Evidence Searching

The following graph illustrates the processes of finding evidences.

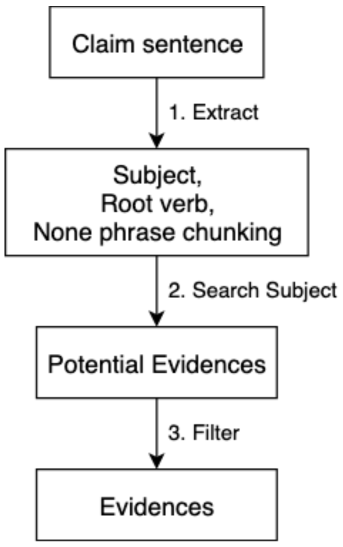

Figure 2: Evidence searching

There are three process for searching. For a given claim sentence, the system firstly extracts the subject, predicate, and noun phrase chunking. Then the subject of the sentence is used to do the searching on the document name index we built before. After this stage, the system will get hundreds of candidate evidences sentences. From these sentences, we set a filter to try to identify the real evidences. This filter will judge if a candidate evidence sentence has the same none phrase chunking or predicate. The system gives a score for a match. The final evidence selection is based on this match score.

2.2 Label inference

For label inference task in this project, we have used the Enhanced Sequential Inference Model (ESIM) with slight modification.

2.2.1 Model structure

The process of the model is shown in figure 3.

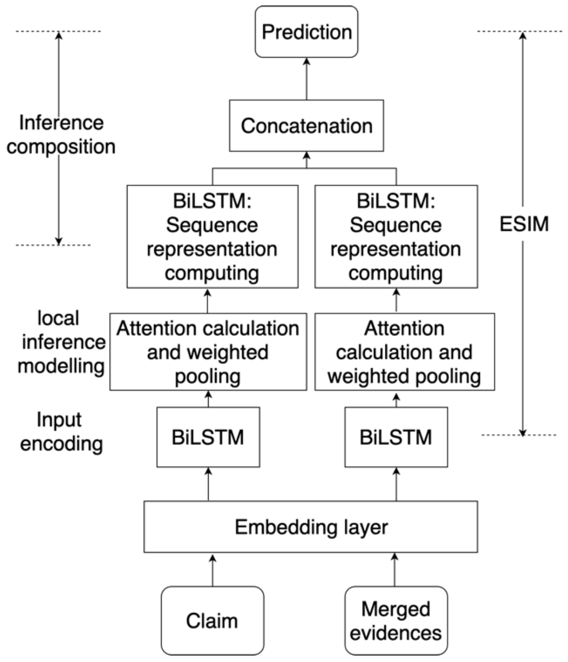
Prediction
Concatenation
Inference composition
BiLSTM: Sequence representation computing
BiLSTM: Sequence representation computing
ESIM
local inference modelling
Attention calculation and weighted pooling
Attention calculation and weighted pooling
Input encoding
BiLSTM
BiLSTM
Embedding layer
Claim
Merged evidences

Figure 3: Structure of the label inference model

There are two parts which will be used as inputs in this model: the merged evidence and the claim. For the evidence sentences, they will be firstly merged with each other to get a single sentence. We have noticed that there are some evidence sentences have longer length than the others in the training set. Such kind of evidences and their claims and labels will be discarded in the training processes. After merging process, only the evidence sentence with less than 500 words will be remained. As to the claim part, this model uses the raw claim as the input.

Embedding: The next layer is the embedding layer. GloVe (Pennington et al., 2014) pre-trained embedding dataset is used in this project. The reason for picking this one is that GloVe is trained with Wikipedia dataset (Wikipedia 2014 and Gigaword 5 with 300 dimensional vectors). This project also use the wiki dataset as the evidence retrieval source. Besides, we believe that use the longer dimensional vectors could represents the meaning of the word better. Therefore, GolVe dataset with 300 dimensional vectors is used. In addition, we set the maximum sequence length for input as 100 for efficiency consideration. The part beyond this range would be dropped since the longer sequence will consume more computing power. Only the word type appeared in the training set will be used to build the embedding matrix.

Input encoding: The next following layer contains two BiLSTM (bidirectional LSTM) (Graves and Schmidhuber, 2005). One of them encodes the claim and the other one encodes relevant evidences. All the input words would be able to represent along with their context (Chen, Zhu, Ling, Wei, Jiang & Inkpen, 2016).

Local inference modeling: With respect to local inference modeling part, attention for each word pair (one word in the claim and the other word in merged evidence) will be calculated, then every word representation will multiply its attention weight (Hanselowski, et al., 2018). Each token would get one single representation with weighted pooling, as a result, there will be two new representation for the original input claim and merged evidence.

Inference composition: In the end, both of these representations will be the input of two BiLSTM for computing the sequence representation along with maximum and average pooling. The output for the BiLSTM will be concatenated together and be used as input for a multi-layer perceptron. The claim with its evidence sentence will be finally classified into three labels: *SUPPORTS*, *REFUTES* or *NOT ENOUGH INFO*.

2.2.2 Training process

As to the training process for the model used, we have trained 10 epochs for the merged evidence sentences with length equal or smaller than 500. Ten percent of the data is used as validation set. To prevent the over-fitting, early-stopping is also used. As a result, the loss and accuracy converge after 180 minutes in Google Collaboratory with GPU runtime.

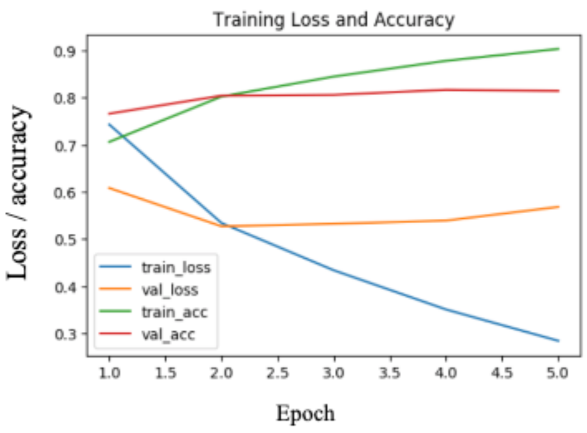

Figure 4: Training accuracy and loss chart

Epoch	Train Acc.	Train Loss	Valid. Acc.	Valid. Loss
1	0.7427	0.7060	0.6082	0.7658
2	0.5341	0.8020	0.5271	0.8039
3	0.4339	0.8444	0.5324	0.8056
4	0.3509	0.8777	0.5392	0.8163
5	0.2844	0.9029	0.5681	0.8139

Table 1: Training accuracy and loss value tables

The training loss and accuracy chart is shown in FIgure 4 and detailed data is illustrated in table 1. We can see that the training and validating set converged at the second epoch. From the third epoch, the model starts to be over-fitting. As a result, the weight of the models is used after the second epoch, which reach the accuracy of 0.8039 for validation set. However, this percentage reflect the accuracy of a sentence with length less than 500 words. For the prediction accuracy with any length sentence is tested in 'devset.json' dataset which will be mentioned later.

3 Result on CodaLab competition

By applying our method, we got the following result:

Task	F1-Score (percent)
Sentence Retrieval	55.5
Document Retrieval	66.9
Label Inference	60.2

Table 2: Result in CodaLab competition

4 Error analysis and future work

4.1 Sentence retrieval

The results prove that the document name is important information to do the index searching. However, after comparing the correct evidence results and our results, we found there are several circumstances this system is not able to deal with.

Multiple evidences support each other: For example, a claim is 'Film A is directed by Australian'. The first evidence is 'Film A is directed by Alice'. Another evidence is 'Alice is an Australian'. In this case, our system is not able to find the second evidence. The potential way to solve this is regarding the evidence as a part of the claim. Then, search the new claim to see if there are any evidences.

Synonyms and antonyms: This IR system assigns the score for each sentence based on the entity matching. However, synonyms and antonyms may in the evidence sentences. They will not be count into the score. A possible way is calculating the word similarity.

Entity chunking: This IR system just use a simple noun chunking and it ignores the special meanings of the word such as organization, person and more. Using high-level named-entity recognition with more pre-defined word categories should improve the performance of this IR system.

4.2 Label inference

To evaluate the model, we use the 'devset.json' dataset to do the testing. We retrieved all evidence sentences which is 100 percent correct. This means the accuracy result reflex the performance of model without the influence of IR system. Besides, this evaluation uses full length sentences without filtering 500 length limits sentences. As a result, the test loss is about 0.964, and the test accuracy is 0.696. This explains why the accuracy in codalab is less than the validation accuracy. If the model set longer word limits than 500 for training, the accuracy could be improved. However, this requires more memory space.

As to the word vector, GloVe gives a decent result since it is pre-trained by wiki articles. However, some other method such as BERT has stronger theoretical support. Compared with GloVe, BERT takes into account the influence of word order. This may improve the performance of the word vector.

In addition to the model, we have tried Siamese BiLSTM model. But it does not perform better than ESIM model we used. After analysing the difference of two model, we believe that inter-sentence attention makes ESIM get better result. This stage is the biggest difference which makes sentences have interaction with each other.

5 Conclusion

This report introduces how we build a fact verification system. Two stages are discussed: IR system and model training. From the result of verification accuracy, the system is evaluated. It proves the correctness of model selection and reveal the drawbacks which could be improved in the future.

Acknowledgments

The GloVe Embedding could be retrieval from: http://nlp.stanford.edu/data/glove.6B.zip

References:

Chen, Q., Zhu, X., Ling, Z., Wei, S., Jiang, H., & Inkpen, D. (2016). Enhanced lstm for natural language inference. arXiv preprint arXiv:1609.06038.

Graves, A., & Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Networks, 18(5-6), 602-610.

Hanselowski, A., Zhang, H., Li, Z., Sorokin, D., Schiller, B., Schulz, C., & Gurevych, I. (2018). UKP-Athene: Multi-Sentence Textual Entailment for Claim Verification. arXiv preprint arXiv:1809.01479.

Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543).