Development of a climate science fact-checking system using BERT

Author
1323782

Abstract

The development of a climate science fact-checking system is presented in this work, leveraging state-of-the-art natural language processing techniques such as the Bidirectional Encoder Representations from Transformers (BERT) model, Dense Passage Retrieval (DPR), and Sentence Transformer architectures. The developed system retrieves and classifies relevant evidence from a vast pool of 1.2 million pieces of evidence, with a limited training dataset. The system's effectiveness is demonstrated by Harmonic Mean score, which evaluate both evidence retrieval and classification performance. The final Harmonic Mean score of this system is 1.844, demonstrating the potential of this system, however still requires further improvements.

1 Introduction

Fact-checking in the realm of climate science is of paramount importance in today's world, where misinformation can proliferate rapidly and have far-reaching impacts. This paper explores the development of a robust fact-checking system aimed at dissecting claims within the field of climate science and corroborating them with evidence from a knowledge base.

The report begins by critically examining the challenges and the resources available to address them. It then investigates methodologies for the retrieval of sentence meanings. The underlying technology of the system is Bidirectional Encoder Representations from Transformers (`BERT`), a state-of-the-art Natural Language Processing technique [1]. In developing the system, several BERT-related models along with the hyper parameters were evaluated, to select the most suitable model. This system uses DistilBERT[1] along with other promising semantic meaning comparison architectures such as DPR[2] and sentence transformer[3] advanced by previous research [2], to extract the most relevant evidence. Then this work further leverages neural networks to classify the gathered evidence.

2 Objectives and Challenges

The primary objective of this research is to design a system capable of accurately identifying and retrieving the most relevant evidence records from a large evidence dataset based on a given claim. This involves understanding the core meaning of the claim and comparing it with each piece of evidence, ranking them based on their similarity scores.

However, a challenge we face is the overabundance of evidence in comparison to the limited number of truth labels available. More specifically, with the given train dataset, there are about 1300 claims, and each of the claims has roughly three related pieces of evidence. Meanwhile, we have more than 1.2 million total pieces of evidence. If we pair the 1.2 million pieces of evidence to each of the 1300 claims, we end up with approximately one billion five hundred sixty million, which is not feasible to train our model. Hence, effective data pruning is essential to this project.

Also, given the variability in the length of each claim and evidence text, conventional NLP processing models like Recurrent Neural Networks (`RNN`) or Long Short-Term Memory (`LSTM`) may not be optimal for this task, as RNNs and LSTMs can suffer from vanishing/exploding gradients, slow training due to their sequential computation, and difficulty in handling long sequences, making them less suitable for tasks requiring understanding of complex, long-range dependencies in text.

[1]https://github.com/huggingface/transformers/tree/master/examples/distillation

[2]https://huggingface.co/docs/transformers/model_doc/dpr

[3]https://huggingface.co/sentence-transformers

3 Approaches

For the language processing, we propose the use of transformer model BERT as our base language processing encoder to retrieve sentence meanings. BERT's attention layer processing allows it to understand the high-level meaning of a sentence, making it ideal for this application.

To further enhance our system's ability to compare and rank the semantic meaning of sentences, we employ the Dense Passage Retrieval (DPR) and Sentence Transformer structures. Sentence Transformer, built on Transformer architecture like BERT, efficiently generates semantically meaningful sentence embedding, simplifying comparison tasks via methods such as cosine similarity. Dense Passage Retrieval (DPR) encodes text into dense vectors to capture semantic similarity, not just keyword overlap, enhancing retrieval tasks in systems where understanding text meaning is crucial. These structures provide us with the ability to generate dense vector representations of sentences, enabling a more efficient and effective comparison process.

For data filtering/pruning, we need a way to prune the evidence set, to extract the most related, yet not the true evidence to each claim. In this way, we significantly reduced the training dataset size, to make it feasible for this research. This is so called In-batch negatives as mentioned in the DPR research paper [3].A graph illustration is shown below in Figure 1.

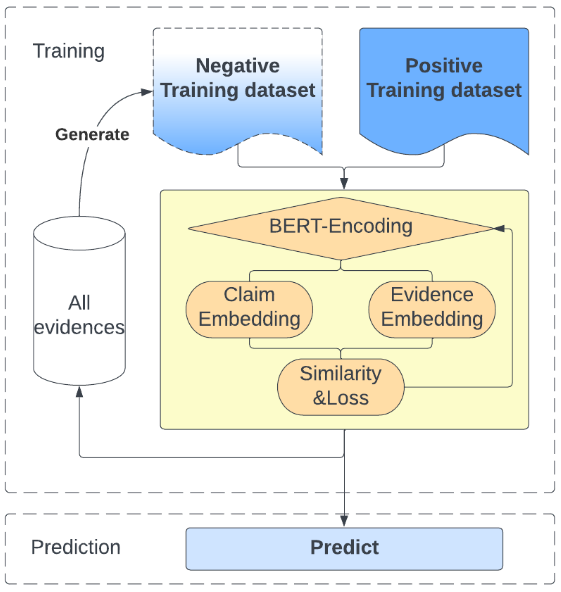

Figure 1: In-Batch-Negatives

We can use the pre-trained model to score and select most related evidence as the negative samples. We can further advance this technique to construct new negative training data before every epoch of training. As every epoch advanced, the model gained more grasp of the semantic meaning of the dataset. This way, we can consistently train our model with higher quality of training data, and use trained models to generate higher quality of negative samples.

In summary, our system leverages BERT, DPR, and Sentence Transformer structures to efficiently identify, compare, and rank relevant evidence records for any given claim.

4 Data analysis

The available data resources encompass two types of datasets: claims and evidence. The claim dataset includes claim IDs, each accompanied by claim text, several claim evidence IDs, and a claim label. The evidence dataset consists of all evidence IDs and corresponding evidence texts, amounting to approximately 1.2 million records.

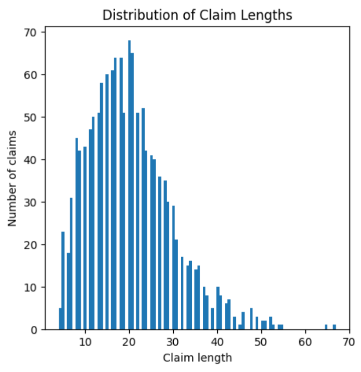

Figure 2: claim number of words

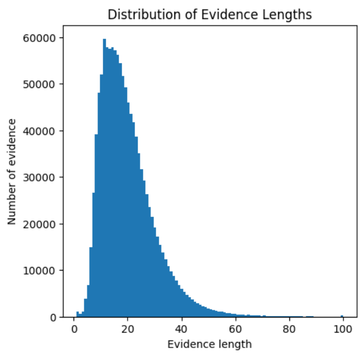

Figure 3: evidence number of words

Before diving into developing the system, we first analyze the data we have. In the claim dataset (including train.json and dev.json), the average claim text length is 21 words, with a low of 4 words and high of 67 words. In the evidence dataset (including train.json and dev.json), the average evidence text length is 19.7 words, with a low of 1 words and high of 479 words. Its distribution is shown in figure 2 and 3. Each claim has an average evidence amount of 3.35, (low: 1, high: 5).

It can be seen that the majority of claim and evidence are with similar number of words. From a manual inspection, we can further assure that both claim texts and evidence texts are in a similar statement format.

This work pre-process the text to lower case, and only keep alpha-numeric chars, comma and periods.

5 Evidence retrieving

5.1 Proposed Architecture

In this paper, we propose to use a simplified Sentence Transformer architecture. The original Sentence Transformer obtain a Siamese BERT-Networks in which both sentence-1 and sentence-2 are passed through its own BERT encoder and a mean-pooling layer, to extract the semantic meaning vectors [4]. Since this work tries to retrieve the most related evidence for a claim and they are in a similar representation, we can use one model to encode both the claim and evidence. Moreover, this work proposes to use the <cls> vector embedding to reduce the computation load while maintain good grasp on sentence meaning.

As for the similarity comparison, this works follows the recommendation in the DPR research, to use dot-product to initially compare the vector similarities, and then use negative log likelihood as the optimized loss function. The proposed architecture in shown in the diagram 4.

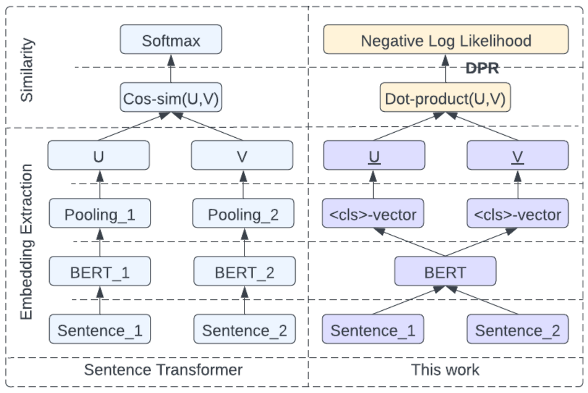

Figure 4: In-Batch-Negatives

Epoch	1	2	3	4	5
BERT-base	0.05	0.07	0.11	0.12	0.14
BERT-large	0.00	0.50	0.08	0.09	0.10
RoBERTa-base	0.04	0.09	0.10	0.13	0.14
RoBERTa-large	0.00	0.04	0.6	0.9	0.11
DistilBERT-base	0.05	0.10	0.14	0.13	0.15

Table 1: Varies BERT model tested

5.2 Experiments

All experiments are conducted in such environment and resources: Python 3.8.3 Transformers 4.29.1 CPU: Xeon CPU @2.20 GHz RAM: 80GB GPU: A100-40GB

This work tested original BERT, RoBERTa, and DistilBERT on a size-reduced evidence set with only 5 epochs and standard hyper parameters. We use the provided f-score calculation method to compare its performance (refer to table 1). Hence, three base BERT-like mode are chosen for candidates. and the baseline is chosen at 0.15 (highest from experiments).

This paper also conducted experiments of max sentence length and number of evidence retrieval. The results showing that max words count of 64 and 4 evidence yield the highest F-score in the test setting.

5.3 Hyper parameters

This work uses Adam Weight optimizer as recommended by Hugging-face and other NLP research papers. The learning rate is set at 2e-5 as followed the recommendation of this paper [5], and handled by a dynamic PyTorch StepLR scheduler.

5.4 Results

DistilBERT-base emerging as the most effective encoder model for evidence retrieving, hence it is chosen as the core encoder. From the chart, its f-score grows quickly in the first 2 epoch, and peaked at 0.225, then struggles to improve its performance any further.

6 Classification

6.1 Proposed Architecture

The model begins by taking the tokenized representations of concatenated claim text and evidence text. These tokens are then fed into the DistilBERT model, which transforms them into rich, contextualized embedding. From this high-dimensional representation, we specifically extract the vector

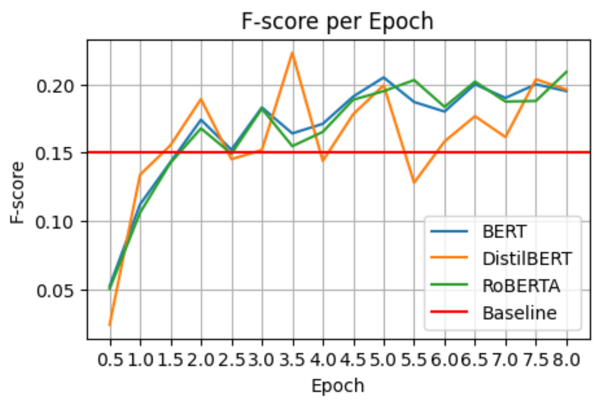

Figure 5: F-score and Loss

corresponding to the <cls> token. This token, standing for "classification", holds the aggregated information from the entire text, making it an ideal input for downstream classification tasks.

The <cls> vector then enters the subsequent neural network. This network is composed of three layers. The input layer has 768 neurons, equal to the dimensionality of our BERT embedding. This is followed by a hidden layer comprising 256 neurons, which serves to further transform and abstract the input data. The final layer is an output layer containing 4 neurons, corresponding to the four possible classes in our task.

The outputs from this layer are passed through a soft-max function. The softmax function ensures that the outputs are normalized to lie between 0 and 1, and their sum equals 1, allowing them to be interpreted as class probabilities. The class with the highest probability becomes the model's prediction.

6.2 Hyper parameters

The loss function proposed in this work is Cross Entropy Loss, with a label smoothing of 0.1. The optimizer is the the same Adam Weight optimizer with weight decay of 1e-4. Learning rate is 2e-5 and handled by a StepLR scheduler with gamma of 0.1.

6.3 Results

The classifier scored a 0.581 Claim Classification Accuracy with given training data.

7 Final Result

During the training stage, the evidence retrieval and classifier combined gained a Harmonic Mean of 0.2891 when deploying the system over the development dataset. During the ongoing evaluation, the system gained a Harmonic Mean of 0.25. In the final evaluation, the system gained a Harmonic Mean of 0.1844. This shows that the system is biased and over fitting, the actual performance is much worse than its training validation performance.

8 Future improvements

One future improvement is to augment data from the given training data, such as using word2vec to replace some words with their synonyms. This helps to generate more training data, also it helps to increase the validation dataset amount, to avoid the chance of over fitting.

The system can adopt the dual-encoder architecture as suggested by DPR and Sentence Transformer. This architecture treats claims and evidence as separate entities during model training. This isolation could alleviate the impact of format differences on the semantic extraction process, leading to more meaningful embedding.

Another potential improvement is the introduction of a pooling layer for semantic meaning extraction. The current system relies solely on the <cls> vector as the sentence embedding. By implementing a pooling layer, we can compute a mean vector from all token vectors within the sentence, thereby potentially capturing more nuances of the semantic content. This approach might lead to more accurate and comprehensive representations, thus improving the overall performance of the system.

9 Conclusion

This project has underscored the efficacy of fine-tuning BERT models in conjunction with well-designed architectures for handling complex natural language processing tasks. While the current system has shown promising results, it has room for improvement, particularly in over fitting and low accuracy. Future efforts should focus on refining our approach to mitigate these limitations. Several strategies, such as adopting a dual-encoder architecture and incorporating a pooling layer for more nuanced semantic meaning extraction, could improve the system's performance. Further empirical testing and experimentation are necessary to evaluate these methods and guide future advancements. Our work thus far affirms the ability of transformer-based models in the quest for sophisticated language understanding.

References

[1] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

[2] Santiago González-Carvajal and Eduardo C Garrido-Merchán. Comparing bert against traditional machine learning text classification. *arXiv preprint arXiv:2005.13012*, 2020.

[3] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*, 2020.

[4] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[5] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. How to fine-tune bert for text classification? In *Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings 18*, pages 194–206. Springer, 2019.