COMP90042: Natural Language Processing
Assignment 3: Evidence Retrieval and Claim Verification for Climate Change Facts (Group 81)

Quynh Phuong Le and Matthew Kuncheria and Jaden Ling

Abstract

Unverified claims can distort public opinion, especially in science, where the general public may lack the expertise to critically evaluate such claims. In this paper, we present a claim verification system that retrieves the most relevant evidence sentences for a given claim and determines, based on those evidences, whether the claim can be supported. The task is divided into two main components: Evidence Retrieval and Claim Verification. We found that our proposed Frozen Encoder Embeddings to Transformer model outperforms the baseline BERT model for claim verification.

1 Introduction

The task to be solved in this paper is a simplified version of the FEVER task (Thorne et al., 2018). The difference in this paper is that, first, the evidences are provided as individual sentences, so the system does not have to search and extract individual sentences from Wikipedia. Secondly, there are four claim labels, “SUPPORTS”, “REFUTES”, “NOT ENOUGH INFO” and “DISPUTED” instead of three as in FEVER shared task. Nevertheless, this task remains difficult to solve in many aspects. Firstly, if the model happens to retrieve very short evidence sentences, it can hinder the claim verification process later, because the model does not have enough information from the evidences to correctly classify the claim. Extending from this, we can see that error propagation is a major issue because the evidence retrieval part will strongly affect the performance of the classification task later. Finally, the additional claim label "DISPUTED" adds more ambiguity to the classification task, which requires a deeper understanding of conflicting information to correctly label this claim.

Other researchers have built state-of-the-art models in the FEVER shared task by implementing BiLSTM architecture and Transformers in their models, also known as Neural Semantic Matching Method (Nie et al., 2018) and Enhanced Sequential Inference Model (ESIM) (Chen et al., 2017). The baseline approach that uses TF-IDF cosine similarity score between claim and evidences also give decent results (Thorne et al., 2018).

In this paper, we first fine-tune a pretrained Sentence Transformer (Reimers and Gurevych, 2019) to get sentence embeddings for the claims and evidences. Then the model retrieves evidences with similarity score higher than a threshold. Lastly, we used a Frozen Encoder Embedding to Transformer architecture to train our classification model.

2 Approach

2.1 Preprocessing

Unicode escape sequences in claims and evidences such as `\u00f1` are decoded to their respective Unicode characters (ñ) to preserve the correct characters. Then, evidence sentences that are less than 10 words long are removed. Based on Table 1, we consider any evidence much shorter than the 25th percentile of the length distribution to be too short, and we believe retrieving these evidences cannot provide sufficient information to verify the claim.

Mean	25%	Median	75%
20	12	18	25

Table 1: Statistical summary of evidence word lengths

2.2 Evidence Retrieval

2.2.1 Sentence-BERT

Sentence-BERT (SBERT) is a Sentence Transformer that is based on the architecture of BERT to compute semantically meaningful sentence embeddings (Reimers and Gurevych, 2019). The model we use is `all-MiniLM-L6-v2` (Wang et al., 2020), which is the best variant of SBERT to balance performance and speed. SBERT is different from

BERT is that, they add a pooling layer on top of the embeddings produced from BERT to get fixed-size sentence embeddings (dim=384 in the case of `all-MiniLM-L6-v2`). Then it trains the fixed-size sentence embeddings in the Siamese/Triplet networks so that similar sentences have embeddings closer to each other (Figure 1).

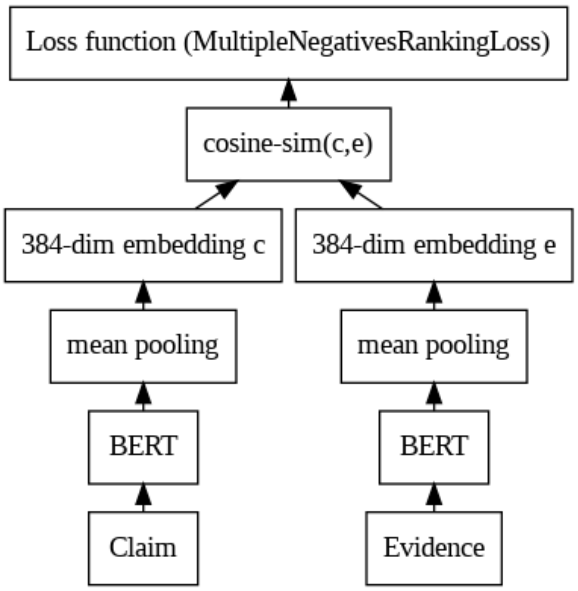

Figure 1: Pipeline of `all-MiniLM-L6-v2` SBERT used

2.2.2 Siamese/Triplet networks

A Siamese network (Figure 1) consists of two identical BERT models with the same initial weights. Then two sentences (claim and evidence) each passes through a BERT model and pooling layer to produce a 384-dimension embedding. Then the cosine similarity (Equation 1) is computed between the claim and evidence embeddings.

$$\text{cos\_sim}(\mathbf{c}, \mathbf{e}) = \frac{\mathbf{c} \cdot \mathbf{e}}{\|\mathbf{c}\| \, \|\mathbf{e}\|} \tag{1}$$

For Triplet networks, same architecture applies but with one additional BERT model (make it a total of 3 BERT models), each for a claim, a relevant and an irrelevant evidence. During training, Triplet networks update the weights to pull similar sentence embeddings together and push dissimilar ones apart. In this paper, the Triplet network is used to minimize `MultipleNegativesRankingLoss`.

2.2.3 MultipleNegativesRankingLoss

Training with `MultipleNegativesRankingLoss` (Henderson et al., 2017) means that given a batch of N positive claim-evidence pairs (ground-truth pairs), for each claim, it takes its correct evidence as positive and the rest N-1 evidences in the batch as negatives (irrelevant evidences) to put into the Triplet network. Then the objective is to minimize:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^{N} \log \left(\frac{\exp\left(\text{sim}(\mathbf{c}_i, \mathbf{e}_i)\right)}{\sum_{j=1}^{N} \exp\left(\text{sim}(\mathbf{c}_i, \mathbf{e}_j)\right)} \right) \tag{2}$$

For each claim $\mathbf{c}_i$, its correct evidence is $\mathbf{e}_i$ and other $\mathbf{e}_j$s in the batch are treated as negatives. The similarity function $\text{sim}(.,.)$ is the cosine-similarity.

This loss function is effective because we want the model to learn better by distinguishing between correct and incorrect evidences, but since we do not have hard-negatives provided, treating others in the batch as negatives is an efficient work-around.

2.2.4 Mean pooling

For an input sentence of length L, BERT models return a matrix of token embeddings $\mathbf{X} \in \mathbb{R}^{L \times d}$, where d is the fixed dimension we define earlier to be 384. Mean pooling is a simple and efficient method that transforms a matrix of token embeddings $\mathbf{X}$ into one vector $\mathbf{s}$ of size d by taking the average across all the tokens (Equation 3).

$$\mathbf{s} = \frac{1}{L} \sum_{i=1}^{L} \mathbf{t}_i \tag{3}$$

where $\mathbf{t}_i$ is the embedding of the i^{th} token from $\mathbf{X}$.

2.3 Claim Classification

For claim classification, we implemented a Frozen Encoder Embeddings to Transformer Model. We compare this proposed model to a standard BERT-Base model as our baseline model.

2.3.1 Motivation and Rationale

Our approach explores whether combining frozen encoder embeddings with a transformer model can yield superior performance compared to directly fine-tuning a BERT model for the claim verification task. While BERT-based models have set benchmarks across various NLP tasks, recent studies, such as those by (Pang et al., 2024), highlight the effectiveness of using frozen transformer blocks as encoder layers, even in non-linguistic contexts. We hypothesise that leveraging frozen embeddings can enhance model performance and generalization by preserving robust pre-trained semantic representations and allowing a lightweight, custom transformer to learn task-specific patterns.

Moreover, excessive fine-tuning of pre-trained models risks disrupting their inherent linguistic knowledge. By freezing the encoder, we preserve semantic integrity, allowing the transformer block to refine the embeddings which significantly boost performance on downstream tasks without extensive fine-tuning (Stankevičius and Lukoševičius, 2024).

2.3.2 Frozen Encoder Embeddings to Transformer Model

We will be using a similar approach in architecture design as shown in (Pang et al., 2024). Although this work primarily focuses on visual tasks, this concept of using frozen transformer blocks as features was adapted to suit this specific NLP tasks. The core idea is to utilise pre-trained sentence embeddings from LLMs while retaining their frozen state to prevent overfitting and computational overhead. We used `all-MiniLM-L6-v2` model as our frozen encoder. This choice was motivated by its efficiency and compact size, it was used as well in the evidence retrieval section.

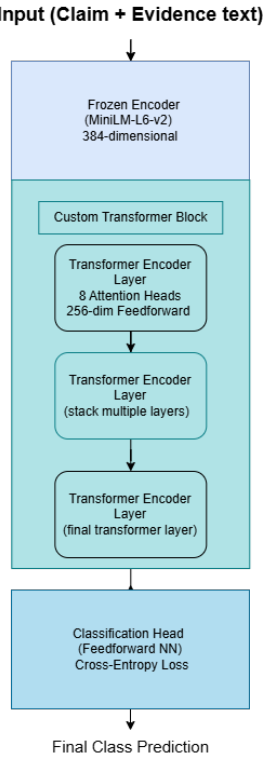

Figure 2: Model Architecture

The model architecture, illustrated in Figure 2, consists of three main components: a Frozen Encoder, a Custom Transformer Block, and a Classification Head. The Frozen Encoder (`all-MiniLM-L6-v2`) generates fixed, 384-dimensional sentence embeddings without weight updates, preserving robust pre-trained semantic representations. These embeddings are subsequently passed through a linear projection layer to match the dimensionality required by the Custom Transformer Block. The transformer consists of stacked encoder layers with multiple self-attention heads and feedforward networks, refining the embeddings while maintaining their semantic integrity. Finally, the refined embeddings pass through a feedforward Classification Head, producing logits used for the claim classification task.

2.3.3 BERT Base Model

As a baseline, we fine-tuned a pre-trained BERT-base-uncased model using the same training data. This model processes raw text inputs and directly learns the classification task from scratch. Similar to the model described previously, we used the Hugging Face API to load and train the baseline model.

3 Experiments

3.1 Evidence Retrieval

We split the training data into a training set (S_{train}) and a validation set (S_{val}), allocating 10% of the data to S_{val} for hyperparameter tuning. All the trainings are done with default learning rate $2\text{e}{-5}$ and batch size equals 32. This batch size is large enough to get sufficient negative evidences while is still safe and stable for GPU memory.

3.1.1 Training epochs

Recall@1 and Mean Reciprocal Rank (MRR) are used to evaluate models with different training epochs. Recall@1 measures how often the correct evidence appears in the top 1 of the retrieved list (Equation 4). MRR measures how high the model ranks the correct evidences (Equation 5). These two metrics are used instead of F1-score because we do not have a score threshold to get the full predicted relevant evidences yet, so Recall@1 and MRR let us evaluate how well the models pull the relevant claim and evidence together.

$$\text{Recall@1} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}\left[rank_i = 1\right] \tag{4}$$

$$\text{MRR} = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{rank_i} \tag{5}$$

where $rank_i$ is the rank of the relevant evidence of claim i and N is the number of claims in S_{val}.

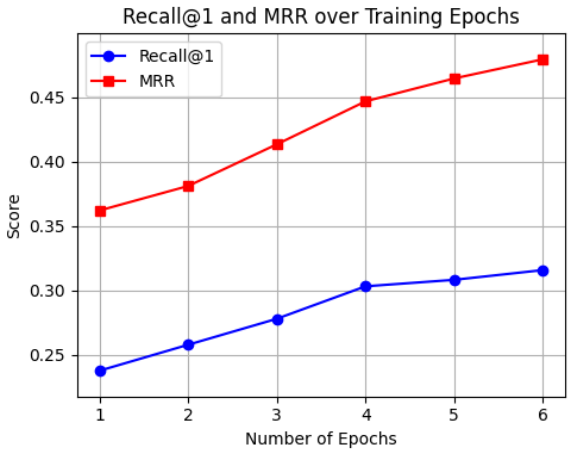

Figure 3: Recall@1 and MRR over Training epochs

Training `all-MiniLM-L6-v2` on S_{train} over different epochs gives results as shown in Figure 3. We can see that the model's improvement starts to slow down beyond 4 epochs. So we decide to use 4 epochs to do actual training on the whole dataset.

When experimenting models with different epochs, the evidence pool that we retrieve from is not the whole set of evidences. Instead, we only retrieve from the pool of all gold evidences in S_{val}. This is because re-encoding the whole set of evidences for different epochs will be very time-consuming, while the pool of gold evidences is a subset of evidences that ensures the claims can still find its correct evidence and gives a more efficient way to evaluate the models.

3.1.2 Score threshold

After training `all-MiniLM-L6-v2` on S_{train} for 4 epochs, the model retrieves evidences for S_{val} whose cosine similarity with the claim is greater than a threshold, i.e., only if cos-sim$(\mathbf{c}, \mathbf{e}) > \tau$, where τ is the similarity threshold, $\mathbf{c}$ and $\mathbf{e}$ are the embeddings of claim and evidence respectively.

Threshold	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
F1	0.02	0.07	0.05	0.01	0.00

Table 2: F1 scores when using different thresholds to retrieve evidences

Table 2 shows that when retrieving evidences for S_{val}, the best threshold that yields highest F1 is 0.6.

3.2 Claim Classification

3.2.1 Experimental Setup

The dataset has already been divided into training and development sets, where the development set was used for evaluating the performance of the model after each epoch. The pre-processing steps included mainly concatenation of evidence sentences for each claim.

We evaluated both models using accuracy and F1 score given the imbalanced nature of the claim verification task. The evaluation was performed only on the development set. Another key thing to note about our setup was that actual evidence given in the development set was used in order to compute the evaluation metrics. This decision was made as only the performance of the classification downstream task should be of concern and the performance of the evidence retrieval section should not play an affect in evaluating the classification.

The Frozen Transformer model required the construction of a custom data collator to ensure that the encoded embeddings were correctly batched for input into the transformer. For the BERT model, the standard data pipeline from the Hugging Face library sufficed.

The fine-tuning was performed using the Trainer API from the Hugging Face Transformers library used in one of the tutorials for both architectures, this enables efficient training with GPU support while also keeping the fine tuning methods the same.

3.2.2 Hyperparameters and Training Procedure

Both models were trained for a maximum of 5 epochs while the learning rate was set to 1×10^{-4}, with a batch size of 16. The default optimiser from the Trainer API was used with a weight decay of 0.01 to regularise the model weights. Training progress was logged every 10 steps, monitoring metrics like loss, accuracy, and F1 score for the evaluation of the development set.

4 Results

4.1 Evidence Retrieval

We fine-tune `all-MiniLM-L6-v2` with 4 training epochs on the whole training dataset and evaluate it on the development set using score threshold 0.6. We achieve F1-score 0.158 on the development set. This is worse than we expected compared to the F1 achieved by using TF-IDF in the baseline (Thorne

et al., 2018), which was 0.1747. This may be due to differences in evidence pool: FEVER uses all of Wikipedia, allowing more retrieval flexibility, whereas this paper relies on a fixed set of individual sentences. We could also use a much larger Sentence Transformer such as `all-mpnet-base-v2` to get higher-quality sentence embeddings.

From Table 3, we found that it is also hard to find relevant evidences for claims with label "NOT ENOUGH INFO", as the average cosine similarity of claims with this label is low across all evidences.

Label	SUP	REF	NEI	DIS
Cos-sim	-0.012	-0.017	-0.005	-0.014

Table 3: Average cosine similarity between claims and all evidence, grouped by claim label

4.2 Claim Classification

The experimental results demonstrate that the proposed Frozen Encoder Embeddings to Transformer model outperforms the baseline BERT model on the claim verification task. As shown in Table 4, our model achieved consistently higher evaluation accuracy and F1 scores compared to the baseline. The model as compared to the bert model showed steady improvement across training epochs, indicating better generalization and stability. We also included the results of the LSTM model (Staudemeyer and Morris, 2019) to show how much better the transformer model performs against a standard neural network model.

Metric	FT	BERT	LSTM
Evaluation			
Accuracy	0.53	0.45	0.29
F1 Score	0.47	0.34	0.26

Table 4: Summary of Model Performance between Frozen Transformer (FT), BERT and LSTM Model

The confusion matrix shown in Table 5 illustrates the model's performance across different classes. The "SUPPORTS" category achieved the highest precision, while the "NOT ENOUGH INFO" and "DISPUTED" classes experienced higher misclassification rates. This outcome aligns with our expectations, as these categories contained claims with minimal or no supporting evidence, making accurate classification more challenging. Overall, the model significantly outperformed the baseline, demonstrating performance of claim verification.

Actual \ Predicted	SUP	REF	NEI	DIS
SUP	54	5	9	0
REF	17	6	3	1
NEI	30	6	5	0
DIS	10	2	6	0

Table 5: Claim Classification for FT Model

4.3 Combined Pipeline

Combining our evidence retrieval and claim classification models, we observe a decrease in accuracy and F1-score, as shown in Table 6. This decline in performance is likely due to compounding errors from each model, resulting in lower overall scores.

Metric	FT	BERT
Evaluation		
Weighted F1 Score	0.35	0.30
Accuracy	0.42	0.38

Table 6: Summary of Model Performance between Frozen Transformer (FT), BERT with pipeline

5 Discussion and Conclusion

The claim classification task demonstrated slight differences in performance between the two models. The Frozen Transformer model achieved higher accuracy and F1 scores across both evaluation setups, illustrating that the use of frozen embeddings was effective in maintaining the semantic representations. However, both models struggled with the DISPUTED and NOT_ENOUGH_INFO classes, as indicated by the confusion matrices. This suggests a need for more refined handling of ambiguous claims and evidence.

A limitation of this study is that the models were trained and evaluated only once, which inherently implies that the run could be 'lucky' and influenced by stochastic factors such as initialisation or batch ordering. Future work should involve multiple training runs with different random seeds to assess the robustness of the findings. Furthermore, while the custom transformer was designed specifically to align with the frozen embedding dimensions, exploring the use of pre-built transformer architectures that can accommodate with the frozen embeddings could be helpful for future reference. They could possibly provide more effective downstream processing and potentially improve the classification performance without much modification

to the current model architecture.

6 Team Contribution

This project was split mainly into two main tasks: Evidence Retrieval and Classification.

Student 1288599 was responsible with the Evidence Retrieval section, while Students 1154499 and 1263521 split the Classification section evenly.

References

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668, Vancouver, Canada. Association for Computational Linguistics.

Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun hsuan Sung, Laszlo Lukacs, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. *Preprint*, arXiv:1705.00652.

Yixin Nie, Haonan Chen, and Mohit Bansal. 2018. Combining fact extraction and verification with neural semantic matching networks. *Preprint*, arXiv:1811.07039.

Ziqi Pang, Ziyang Xie, Yunze Man, and Yu-Xiong Wang. 2024. Frozen transformers in language models are effective visual encoder layers. *Preprint*, arXiv:2310.12973.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Lukas Stankevičius and Mantas Lukoševičius. 2024. Extracting sentence embeddings from pretrained transformer models. *Applied Sciences*, 14(19):8887.

Ralf C. Staudemeyer and Eric Rothstein Morris. 2019. Understanding lstm – a tutorial into long short-term memory recurrent neural networks. *Preprint*, arXiv:1909.09586.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Preprint*, arXiv:2002.10957.