Automated Verification of Climate Science Claims: A Hybrid Approach Using BM25 and Cross-Encoder

THU1PM_Group2

Abstract

In this age where the internet is so advanced and everyone can easily share information on the web, it has become crucial to verify the accuracy of the information. Our research aims to develop an automated fact-checking system specifically for climate science. This system checks the reliability of claims related to climate science and finds supporting evidence passages from a knowledge base. Our main approach has three steps: First, using the BM25 model to calculate the similarity between claim text and evidence, then using a cross-encoder to rerank the evidence to find the most relevant ones. Finally, the label of the claims is determined based on the available evidence. Our approach provides better training results while efficiently utilizing computational resources.

1 Introduction

Climate change has become a critical global issue. However, this has led to the proliferation of unsubstantiated climate science claims on social media, polarizing public opinion. This underscores the urgent need for fact-checking such claims.

Our project aims to develop an automated fact-checking system for climate science. For each climate-related claim, the system will retrieve relevant evidence from a large knowledge base and determine whether the claim is supported. The system will classify the claim into one of four categories: SUPPORTS, REFUTES, NOT_ENOUGH_INFO, or DISPUTED. This system aims to enhance information accuracy and support the formation of well-informed public opinions and policies on climate change.

2 Approach

2.1 Exploratory Data Analysis

Before the development of the system, we first performed a detailed exploratory data analysis (EDA). Our dataset consists of three parts: a training and validation set of labeled claims, a test set of unlabeled claims, and a knowledge base containing many evidence passages.

We analyzed the label distribution of the training and validation datasets. In both datasets, the number of support labels is the highest, followed by claims with not enough info, and the proportion of refutes and disputed is relatively low. We also analyzed the average length of claim and evidence texts. The results show that the average length of claim texts in the training set, validation set, and test set is around 120 words, while the average length of evidence texts is 119.51 words. This suggests that we need to choose a suitable text processing method to enable the model to learn the features better.

Additionally, we counted the number of evidence associated with each claim. In both the training and validation sets, there are 1-5 evidence associated with each claim, claims with five associated evidence have the highest percentage, while the distribution of others are similar.

Furthermore, by performing word frequency analysis on the corpus, we found that the dataset contains nearly 800,000 unique unigrams. Such a large number of words poses a significant challenge to computational efficiency. Therefore, we need to obtain words that are both informative and discriminative through specific methods, thereby optimizing our algorithm.

The next section will detail our main approach, compare it with other methods, and explain why our chosen method is superior.

2.2 main approaches

Since the training data is too large (over 1 million evidence), it is difficult for us to use neural network in the initial training due to the GPU memory restriction and computational time, so we decided to compute the similarity between claim and top-k evidence, then use neural network to rerank the top-k evidence and select true evidence, which is an efficient way to save computational resources and provide better training results. Sothe main approach has three steps: Firstly, Using the initial ranking model to calculate the similarity between claim and evidence. Then use the Rerank model to find the True evidence. Finally, determining the label of a claim based on available evidence.

2.2.1 pre-processing

Before retrieval, it is very important to maintain the consistency of word forms and parts of speech. This can help reduce vocabulary complexity and noise, making important information more prominent. The main steps shows below:

- Convert all text to lowercase: Case does not change the semantics, and it allows for easier comparison of vocabulary.

- Remove stopwords and punctuation: This reduces the dimensions of data and noise, highlighting more important information.

- Lemmatization: This helps identify the same word in different tenses or aspects.

2.2.2 initial model: BM25

The BM-25 is first mentioned by SE Robertson K Spärck Jones, It calculates relevance score between different documents. Compared with TF-IDF(Term Frequency-Inverse document frequency) or BoW(Bag of words), It introduced two parameters: length normalization(b) and term frequency saturation(k1). Length normalization normalizes the scores of long documents. (Robertson and Walker, 1994)

The BM25 scoring function for a document d and a query q is defined as:

$$score(d, q) =$$

$$\sum_{i=1}^{n} IDF(q_i) \cdot \frac{f(q_i,d)\cdot(k_1+1)}{f(q_i,d)+k_1\cdot\left(1-b+b\cdot\frac{|d|}{avgdl}\right)}$$

where:

- $f(q_i, d)$ is the term frequency of the term q_i in the document d.

- $|d|$ is the length of the document d (in words).

- $avgdl$ is the average document length in the text collection from which documents are drawn.

- k_1 and b are hyper-parameters, usually chosen, in the absence of advanced optimization, as $k_1 \in [1.2, 2.0]$ and $b = 0.75$.

- $IDF(q_i)$ is the inverse document frequency of the term q_i and is given by:

$$IDF(q_i) = \log\left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1\right)$$

- N is the total number of documents in the collection.

- $n(q_i)$ is the number of documents containing the term q_i.

By tuning b, the scores can be made to depend not only on the word frequency/Inverse document frequency but also on the length of the document, avoiding the possible scoring bias of long documents. Also, Adjust the parameter will change the importance of term frequency, more details will be shown in the experiment section.

2.2.3 cross-encoder + Rerank model

However, the relationship between the claim and evidence is not always apparent. For example, temperature & degree, carbon dioxide & CO2. The hidden relationship between tokens may need a complex model to understand. In this task, we decided to use Transformer to capture the relationship between the claim and evidence, Before processing, we decided to use the cross-encoder. The Cross-Encoder processes a pair of sentences by concatenating them with a separator and passing them through a Transformer model. The structure can be represented as follows:

$$"Claim\ text" + [SEP] + "Evidence\ text"$$

In this situation, evidence retrieval becomes a supervised binary categorization problem, where the label is 1 if the claim text is correctly paired with the correct evidence, and 0 otherwise. With this text structure, the attention mechanism can not only capture the relationship inside the claim or evidence text but also consider the attention matrix relationship between the claim and evidence.

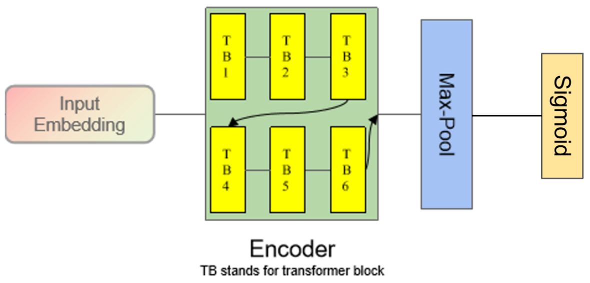

Figure 1: Rerank model based on transformer.

As the graph(Figure 1) shows, We implemented a custom cross-encoder model. The model accept the concatenated claim and evidence text, and it comprises word embeddings with added positional embeddings, followed by a transformer encoder block. The output layer is a sigmoid layer that processes the global max pooling result of the encoder.

2.2.4 fact-checking model

Once we select the top k predicted evidence, we concatenate the claim with all top k predicted evidence with <SEP> between each evidence or claim. We then input the concatenated sentence into a model that has the same architecture as our reranking model, but replaces the output layer with softmax that map to 4 categories of claim label. Figure 2 shows the final structure of the fact-checking system.

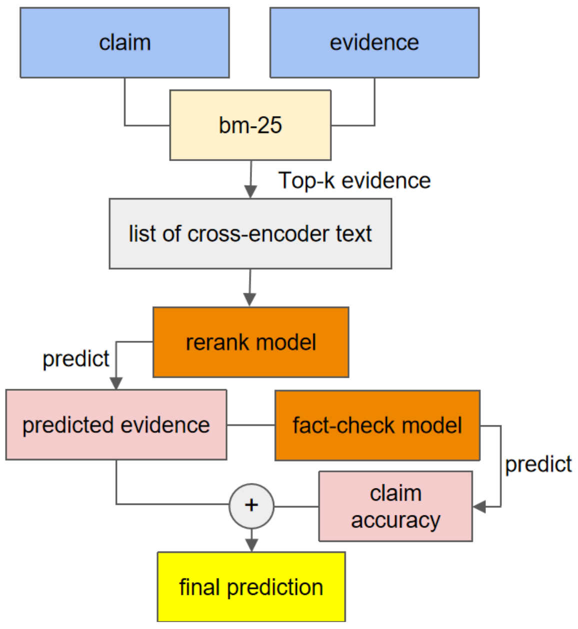

Figure 2: final architecture

2.3 Other approaches: bi-encoder

The Bi-Encoder architecture utilizes two independent encoders to process two distinct inputs, namely the evidence and the claim, each producing a high-dimensional vector. These vectors can then be compared using cosine similarity to facilitate evidence retrieval. However, this setup often results in a significantly low recall rate. This issue primarily arises due to biencoder hard to capture more contlex since it has less complex interaction between two inputs. Consequently, when attempting to convert all evidence into vector form, the model may struggle to effectively transform the evidence and claim into closely matching embedded vectors.

2.4 Other approaches: Time series model

When we used the cross-encoder structure, we tried time series models such as RNN and LSTM. However, the results were never good, and we guessed that it might be because the number of tokens in the cross-encoder structure exceeded the length that could be handled by the time series model. This leads to gradient vanishing.

3 Experiments

3.1 Evaluation method

3.1.1 initial ranking

similarity calculation	Recall		
	Top3	Top20	Top100
TF-IDF	0.110	0.230	0.430
BM25 (Pre-Tuning)	0.140	0.323	0.506
BM25 (Post-Tuning)	0.146	0.362	0.516

Table 1: Recall Comparison

To further extract evidence in the rerank model, we need to ensure that the initial model captures enough evidence (high recall). Table 1 shows the recall of evidence in the development set for top-k(k=3,20,100) similarities calculated by TFIDFIt calculates the cosine similarity of two list of TFIDF value and BM-25. It is obvious that BM25 has a significant improvement in top-k recall compared with TF-IDF and there is a slight difference between pre-tuning and Post-tuning BM25.

However, Although the top 100 BM25 has the best recall, it might face data imbalance when it goes to the rerank model. As the EDA mentioned before, the number of evidence for each claim is between 1-5, with the higher recall, the proportion of false evidence also increased. So we need to trade off the balance between recall and top-K. In the result section, we will try to provide the best performance based on the combination of different top-k and rerank models.

3.1.2 Cross-encoder Ranking

In evaluating the cross-encoder ranking results, we sorted the sigmoid output probabilities of claim-evidence pairs for each claim from high to low to finalize the reranking process. We also tested the ensemble method, which concatenates the top k evidences from BM25 and the top k evidences from the cross-encoder. Although the cross-encoder's results can be monitored by metrics like TP and

FP, we found the model was too strict in classifying true evidence. Therefore, we continued using the F-score to evaluate the top sorted probability evidence.

3.2 Experimental details

For the tuning of initial model, we used grid-search to find evidence retrieval with different k1 and b values in the top 200. Since computing the BM25 score for each claim is a task of considerable complexity, we chose to perform two 5*5 grid-search(5*b value and 5*k1 value).

The hyper-parameter of pre-tuning BM25 is 1.2(k1) and 0.75(b), the post-tuning BM25 is 0.5 and 0.85. This means the importance of term frequency needs to be lower and the importance of document length needs to be higher. This confirms our conjecture in EDA that term frequency may be somewhat disruptive to finding specific key phrases.

In the training set, we included evidence not retrieved by BM25 to expose the model to more label 1 data, as the top BM25 evidence is often incorrect (label 0). In contrast, the development set only included top evidence from BM25, mimicking the test set where ground truth labels for evidence are unavailable.

When configuring our model, we used BERT as a reference. In BERT base (L=12, H=768, A=12, Total Parameters=110M), L represents the number of layers (i.e., Transformer blocks), H is the hidden size, and A is the number of self-attention heads (Devlin et al., 2019). Due to device limitations, using 12 layers was too time-consuming, so we reduced L to 2 and selected H = 1024 and A = 4 in our custom model. The model was trained using the Adam optimizer with a learning rate of 1e-5 for 10 epochs, and the claim classification used the same configuration. Although the experiment tested could run on a Colab T4 free version GPU and allocated up to 4.5GB GPU Memory with batch size 32, each epoch took approximately no more than 2 minutes.

4 Results

4.1 Retrieval Results

We compared the top 3 reranked results from the top 20 BM25 retrieval evidence with directly predicting the top 3 BM25 retrieval evidence. The reranked results did not outperform BM25's direct predictions, likely due to imbalanced training data.

Evidence Source	Dev	Codalab
Top4 BM25	0.135	NA
Top3 BM25	0.130	0.078
Rerank Top 4 from top10 BM25	0.112	NA
Rerank Top 4 from top20 BM25	0.086	NA
Top3 Bm25+Top1 rerank	0.147	0.089
Top4 Bm25+Top1 rerank	0.147	0.095
Top5 Bm25+Top1 rerank	0.140	0.091

Table 2: Evidence Retrieval F-score

Narrowing the search range to the top 10 BM25 retrieval evidence improved reranked results in the development set but still did not surpass BM25 alone.

We hypothesized that combining BM25 and the transformer's reranked results could be better than using BM25 alone. This is because the transformer can rank evidence that is semantically related. The ensemble approach did improve the development set results, as shown in Table 2, but there were significant differences between the development and test sets. While ongoing Codalab evaluations showed similar results between sets, the final evaluation dropped below an F-score of 0.1, indicating possible dataset-specific biases.

Although the ensemble results achieved a top ranking on Codalab, this came at the cost of precision, with more than half of the claims having fewer than three related evidence pieces. Therefore, we need a more robust cross-encoder reranking structure. The cross-encoder can identify related evidence not similar in term frequency, potentially surpassing BM25 alone. We believe pretraining the cross-encoder on the 1M+ evidence corpus could improve its performance. As shown by RoBERTa (Liu et al., 2019) and BERT (Devlin et al., 2019), the pretraining stage is crucial for achieving superior performance.

4.2 Claim Classification Results

Evidence Source	Dev	Codalab
Ground Truth Evidence	0.623	NA
Top3 Bm25+Top1 rerank	0.461	0.454
Top4 Bm25+Top1 rerank	0.448	0.428
Top5 Bm25+Top1 rerank	0.441	0.324

Table 3: Claim Classification Accuracy

Our initial attempts to predict claim labels based on predicted evidence yielded disappointing results

(Table 3). Despite the model's unpredictable behavior, we noticed that inputting fewer predicted evidence pieces improved performance (Table 3). This led us to hypothesize that incorrect evidence inputs were affecting claim classification.

Experiments using ground truth evidence confirmed that incorrect evidence was a significant factor, as the model showed acceptable performance under these conditions (Table 3). This aligns with other research on fact verification systems, which indicates that a low recall rate can impact claim label accuracy (Wang et al., 2019). Although our recall is relatively higher than precision, this increase comes at the cost of precision, highlighting the need for improvement.

A compromise solution remains elusive, as the model's behavior persists without significant improvement in retrieval stage precision. While filtering low-probability evidence might help, we believe human intervention in post-processing could introduce overfitting and reduce generalization. Thus, developing a more robust retrieval mechanism is crucial for this research.

4.3 Impact of model configuration

In comparison of different model configuration, we treat the top 4 re-rank result from cross-encoder model as our criteria. Initially, we observed that the embedding dimension significantly influences the re-ranking results on the development dataset, with a higher dimension correlating with improved F-scores (Table 4). We then fixed the embedding dimension at 1024 and varied the number of heads. Surprisingly, increasing the head count from 2 to 4 improved the F-score, but further increases showed no additional gains (Table 5).

Embedding dimension	**Dev F Score**
1024	0.112
768	0.102
512	0.092
256	0.090

Table 4: Embedding dimension impact(head num = 4)

Number of heads	**Dev F Score**
8	0.107
6	0.104
4	0.112
2	0.103

Table 5: Attention head impact(Embedding dimension = 1024)

This phenomenon aligns with findings from the study "Low-Rank Bottleneck in Multi-head Attention Models," (Bhojanapalli et al., 2020) which suggests that increasing the number of heads, while keeping the embedding dimension constant, reduces the dimensionality per head, creating a 'Low-Rank Bottleneck.' This reduction likely explains our experimental observations. Since while more heads can theoretically capture more diverse information, the reduced dimensionality per head restricts what each head can capture. Consequently, avoiding this bottleneck would require larger embedding dimensions (Bhojanapalli et al., 2020). However, it is not realistic to enhance model performance by increasing the embedding size due to device limitations in this project. Research on ALBERT suggests a method to reduce overall parameters while potentially increasing the embedding size.(Lan et al., 2020) Given that our architecture ties the transformer block's embedding size to the word piece embedding size like BERT (Lan et al., 2020), we could use a smaller word piece embedding size and project it to a higher-dimensional embedding size in the transformer block. This approach could increase the transformer block's embedding size while reducing parameter usage. Additionally, cross-layer parameter sharing could allow us to deepen the network without increasing the parameter size (Lan et al., 2020), providing a significant area for future experimentation on reducing parameter numbers and balancing embedding size with the number of heads.

5 Conclusion

In this paper, we explored a self-implemented fact verification system for climate science. We examined the dataset, identifying key aspects, and experimented with various methods for the initial ranking stages. Our system utilized a cross-encoder structure to further refine the top evidence from the initial rankings. Final predictions were made based on an ensemble of cross-encoder and BM25 rankings. We discussed the results of different methods, analyzed the potential impacts of the ensemble approach and model configuration, and highlighted potential ways to build a more robust model for retrieval and classification stages.

6 Team contributions

Weihao Zhao is mainly responsible for the initial rank, including the selection of algorithms such as BOW, TFIDF and BM25.

Xinyuan Qiao is mainly responsible for reading rank related papers, giving opinions on the optimization model, and part of the tuning (e.g. grid-search in BM25).

Tianhao Liu is responsible for all the experiments, including the selection of initial rank and rerank models, tuning, and updating the progress of the experiments in real-time and where the difficulties are encountered.

7 References

References

Srinadh Bhojanapalli, Chulhee Yun, Ankit Singh Rawat, Sashank J. Reddi, and Sanjiv Kumar. 2020. Low-rank bottleneck in multi-head attention models. *Preprint*, arXiv:2002.07028.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *Preprint*, arXiv:1810.04805.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. *Preprint*, arXiv:1909.11942.

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. *Preprint*, arXiv:1907.11692.

Stephen E Robertson and Steve Walker. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR'94: Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, organised by Dublin City University*, pages 232–241. Springer.

Peiyi Wang, Lixia Deng, and Xiujun Wu. 2019. An automated fact checking system using deep learning through word embedding. In *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 3246–3250.