COMP90042 - Final Project Report
Automated Fact-Checking of Climate Claims

Tanzid Sultan
University of Melbourne

Abstract

Stopping the spread of climate science misinformation, which is rampant all over the internet, is crucial for ensuring informed public decision-making and policies. The in-feasibility of manual fact-checking by human experts necessitates automation. We explore a machine learning approach to building an efficient and accurate automated fact-verification system for climate claims. Our two-stage approach includes separate retriever and classifier components. We experiment with a vanilla BM25 retriever and enhance it with a transformer-based re-ranking procedure. Additionally, we experiment with a transformer-based classifier and are able to improve its performance with a multi-task pre-training strategy. Despite our very modest dataset size and small model scale, we demonstrate that transformer-enhanced retrieval and multi-task pre-training can improve performance.

1 Introduction

With the rise of social media and the prevalence of independent online media outlets, there is ever increasing risk of the spread of misinformation. This is particularly true in the case of misinformation regarding climate change which is a topic of much contention. While some misinformed claims are debunked by climate science experts, the sheer volume of such claims necessitates automated systems to swiftly verify these claims, substantiated by scientific literature and public discourse.

The claim verification task involves two stages: finding documents containing relevant information about the claim, and then using these documents to verify the claim. This is an inherently challenging task for an automated system because it requires natural language understanding (NLU). Recent advances in transformer-based learning techniques for NLP, such as Large Language Models (LLMs), have enabled automated systems to solve NLU tasks rivaling human-level performance.

The **two-stage** approach of retrieval and extraction for knowledge-intensive NLU tasks has been extensively researched, especially in the context of Open-Domain Question Answering (Huang et al., 2020). This general framework, known as the **reader-retriever model**, involves finding relevant documents and extracting potential answer spans, and has been successfully applied in automated question-answering and fact-verification systems (Zhang et al., 2023, Guo et al., 2022). Datasets have been developed for training machine-learning based fact-verification systems across various domains, with a smaller focus on climate science claims, such as the Climate-FEVER dataset of Diggelmann et al. (2020).

Most state-of-the-art (SOTA) fact-verification systems utilize the reader-retriever approach and employ large pre-trained transformer-based language models such as BERT (Devlin et al., 2019) which are ideally the backbone for any modern NLU task. The retrieval stage uses either traditional statistical techniques like **BM25** (Robertson et al., 1995), which creates sparse document representations, or pre-trained transformers for dense vector representations, such as **dense passage retrievers** (DPR) (Karpukhin et al., 2020). BM25 is simple and efficient, however it relies on direct word matches and so retrieved documents may not be semantically aligned with the query. DPR directly leverages pre-trained transformer knowledge for semantic content search. An alternative approach, **monoBERT**, uses BM25 for initial retrieval and a pre-trained transformer for re-ranking, improving document recall significantly (Nogueira and Cho, 2019, DeHaven and Scott, 2023). Unlike DPR, monoBERT encodes a single sequence containing a claim concatenated with a document. This allows the self-attention mechanism to learn complex relationships between the two sentences leading to superior performance over DPR for the document relevancy scoring task (e.g. Yates et al., 2021).

The claim classification stage typically employs one or more pre-trained transformers for multi-class claim veracity classification. Techniques vary from simple concatenation of all retrieved evidence documents with the claim (Diggelmann et al., 2020), to pairing each evidence document with the claim for individual classification and aggregation (Soleimani et al., 2020). More complex techniques include multi-hop reasoning (Ma et al., 2023) and kernel-graphs for evidence aggregation (Liu et al., 2020). Some approaches also utilize joint multi-task training for both retrieval and classification stages. Most of these SOTA approaches for automated fact verification have been shown to achieve high performance as evident from the FEVER (Thorne et al., 2018) and KILT (Petroni et al., 2021) benchmarks.

Our **goal** for this project is to design and implement an accurate and efficient automated fact-verification system specifically for climate science claims. Our design choices draw heavily from the aforementioned SOTA methodologies and innovations from the research literature. Our system follows the reader-retriever framework with two separate stages for retrieval and claim classification, as shown in Figure 1.

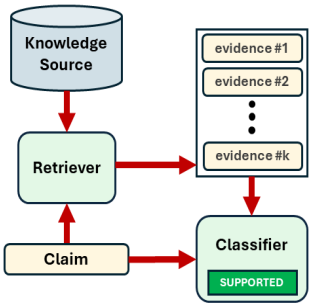

Figure 1: Overall two-stage system architecture.

We have chosen the monoBERT method in the retrieval stage, motivated by the distinct advantages offered by this method, as previously discussed, and also due to it's simplicity and efficiency. For the classification stage, we use a simple approach where the claim and relevant evidence passages are concatenated into a single sequence which is then encoded using a transformer followed by softmax classification on the embedding vector of a special [CLS] token. This approach of jointly encoding the claim and relevant evidences could allow the self-attention mechanism to learn useful correlations and interactions between the sentences and therefore lead to improved performance compared to independently encoding the sentences.

2 Approach

In this section we will describe our overall system design approach. At a high level, our system has two main components: (1) a retriever and (2) a claim classifier. Given a claim c, the retreiver's task is to find the *top-k* most relevant evidence document from a static knowledge source S. We denote the set of retrieved documents as $D_k = \{d_1, d_2, ..., d_k\} \subset S$, with k being a fixed hyperparameter. Then given D_k, the task of the claim classifier is to compute a conditional probability distribution $p(y_c|D_k)$ for the claim belonging to one of four possible classes, $y_c \in$ {SUPPORTS, REFUTES, NOT_ENOUGH_INFO, DISPUTED}.

Current SOTA systems use large transformer models like BERT, pre-trained on vast text corpora, for both retrieval and classification stages (Guo et al., 2022). These models learn rich language representations, including context-dependent word meanings, grammar, and syntax. Fine-tuning a pre-trained transformer on a smaller task-specific corpus effectively transfers knowledge from the large pre-training corpus, leading to substantial performance improvements without additional pre-processing or feature extraction (such as POS Tagging, NER, parsing, etc.). Therefore, we have designed a multi-task pre-training strategy for own custom BERT model.

2.1 Transformer Multi-task Pre-training

Our custom BERT, implemented in PyTorch, follows similar architecture and pre-training of Devlin et al. (2019) with similar hyper-parameters. Due to restrictions, our model is trained solely on our dataset of evidence documents and claims. Unlike the original BERT, which was pre-trained on **masked language modeling** and **sentence entailment** tasks, we added a third task: claim label classification. This helps the model learn correlations between related sentences, crucial for claim classification. For sentence entailment and claim classification, separate linear layers map the [CLS] token embedding to output logits, and cross-entropy loss is computed. The input pairs are either (claim, evidence) or (evidence, evidence), with positive or negative labels assigned based on the gold evidence list. Negative pairs are created by randomly selecting evidences from the knowledge source. We also further modified BERT's token masking scheme to

mask spans of 1-3 tokens, to improve the model's ability to learn phrase-level structures and dependencies.

We also implemented and trained our own **WordPiece** sub-word tokenizer on our dataset. Due to the small corpus size and noise, we found that extensive training (i.e. vocabulary sizes up to 60,000) resulted in very short sub-words mostly 3-4 characters long, and many common English words failed to emerge. This led to excessively long tokenized sequences. To address this, we augmented the vocabulary by adding the 5000 most common words from our corpus to the 15000 tokens learned by the tokenizer. This approach preserves common full-word tokens, resulting in shorter sequences while handling out-of-vocabulary words with smaller sub-words.

2.2 Retriever: monoBERT

Our retriever component is based on the monoBERT approach of Nogueira and Cho (2019). Given a claim sentence c, we first use a BM25 model to compute similarity scores between the claim and every document in the knowledge source. Then we keep the top-k highest scoring documents $\{d_1, ..., d_k\}$. Then for each top-k document d_i, we prepare an input sequence containing both the tokenized c and d_i separated by the special [SEP] token of BERT as shown below:

$$\text{input\_seq} = [[\text{CLS}], c, [\text{SEP}], d_i]$$

We encode this input sequence using our pre-trained BERT model and extract the [CLS] token embedding. This vector feeds into a 3-layer feed-forward network with a sigmoid-activated output neuron, producing a relevancy score between 0 and 1 for the document given a claim. We re-rank the **top-k** documents based on these scores and retain a smaller subset of **top-k'** documents with the highest scores. This re-ranking is most effective with large k and much smaller k' (e.g., k=1000 and k'=5). Figure 2 illustrates this process, showing how the initial BM-25 rankings of gold evidences are significantly improved after re-ranking, ideally placing gold documents closer to the top.

Our BM25 implementation follows Jurafsky and Martin (2024), with an inverted index used for sparse vector representations of claim text and documents, similar to TFIDF. Unlike TFIDF, BM25 additionally modifies term frequencies to account for document length, improving performance. To

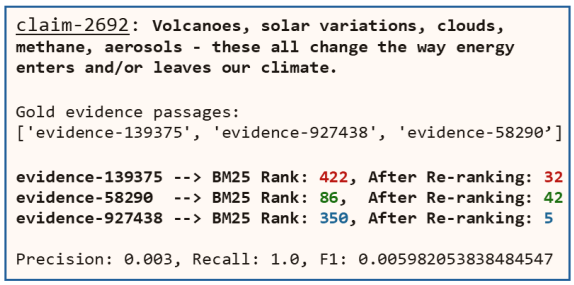

Figure 2: monoBERT retrieval with $k = 500$. Note the difference before and after re-ranking.

fine-tune the re-ranking BERT model, we add a linear "classifier-head" layer that maps the [CLS] embedding to two output logits, followed by a binary cross-entropy loss. The task dataset consists of (claim, document) pairs, labeled positive if the document is in the gold evidence list and negative otherwise. Negative samples include random and "hard negatives" ranked highly by BM25, which can potentially help the re-ranker learn to identify gold passages better.

2.3 Claim Classifier

Our classifier model is implemented by attaching a classifier head to our pre-trained BERT, consisting of a linear layer that maps the [CLS] embedding to 4 output logits, using a cross-entropy loss function to obtain a probability distribution over claim classes. Both the BERT model parameters and the classifier head are fine-tuned on the training set. During training, the input sequence includes the claim and concatenated gold evidences separated by [SEP] tokens. During inference, the sequence includes the claim and the **top-5** relevant evidences returned by the retriever. We use top-5 as that is about the most we could fit within the transformer's block size. If the combined length exceeds the block size, random portions are cropped to fit.

2.4 Baseline

To measure the efficacy of our approach, we benchmarked our system against a simple baseline with retriever and classifier components. The baseline retriever uses plain BM25 without BERT re-ranking. The baseline classifier is identical to our main classifier, except it uses an untrained BERT model with randomly initialized weights instead of fine-tuning on pre-trained weights. This baseline allows us to study the effects of multitask pre-training for our BERT model and establish the superiority of monoBERT retrieval over vanilla BM25.

3 Experiments

Before training, we cleaned claim texts and evidence passages by converting them from Unicode to ASCII and removing URLs. This reduced noise and the number of distinct characters, improving retrieval performance, particularly for BM25 which relies on exact keyword matching.

3.1 BERT Pre-Training

Hyperparameter	Values	Best Value
embedding dims	128, 256, 512	512
self-attention heads	8, 12, 16	16
encoder layers	2, 4, 8	8
batch size	16, 32, 64	32
num. epochs	n/a	150
learning rate	10^{-4}	n/a

Table 1: BERT hyperparameter exploration

For pre-training our BERT model, we used PyTorch's AdamW optimizer due to it's robust convergence, stability and learning rate insensitivity along with a *CosineAnnealing* scheduler for faster convergence. We chose learnable positional embeddings over static ones. We also use a block size (i.e. maximum input sequence length) of 128 as larger values make training time too long. Due to long training times even with GPUs, we explored only a limited range of hyper-parameters (see Table 1) and selected the best-performing combination on the development set.

Evaluation Method: During pre-training, we monitored loss, accuracy and precision and recall for the entailment positive class, averaging these metrics across the training and development splits. Training was terminated after 150 epochs (about 22 hours) when improvements plateaued across all metrics.

3.2 Retriever and Classifier

We applied additional pre-processing and word normalization to the BM-25 training corpus, including lower-case folding, stop-word removal, and lemmatization using the Porter Stemmer and found that this significantly improved performance across all metrics, as BM25 relies on exact keyword matching. We fine-tuned the re-ranking model for 4 epochs and the classifier BERT model for 3 epochs using the same optimizer and scheduler that were used during pre-training, and using early stopping, i.e. we train up to the number of epochs where validation loss stops decreasing. We also found that creating imbalance in the re-ranking task dataset by having 12 times more negative samples than positive samples improved the re-ranking performance. We also experimented with two **query expansion** strategies including word synonym substitution using NLTK's WordNet and back-translation using the Helsinki-NLP Opus-MT model from Huggingface. Even though this leads to a modest 1% increase in average F1-score, we excluded query expansion from our final model due to the high computational cost.

Evaluation Method for Retriever: We compare two retriever models: (i) baseline BM25 and (ii) monoBERT (BM-25 followed by BERT re-ranking). For each model, we retrieved top-k documents for a wide range of k values for claims in the dev set and computed the following evaluation metrics: average Recall and F1 scores which measures how well the model is able to retrieve actual relevant/gold documents and Mean Average Precision (mAP) which measures the ranking quality, i.e. whether the gold documents are actually ranked higher or not.

Evaluation Method for Classifier: We compare two classifier models: (i) baseline BERT (not pre-trained) and (ii) fine-tuned BERT (with pre-training weights). We use overall classification accuracy on the dev set (fraction of correctly classified instances) as the evaluation metric. During training, each claim is classified based on concatenated gold evidence passages, while during inference, it is classified based on the top-5 retrieved evidences. We conduct **3 experiments** to measure classification accuracy: (1) classifying using concatenated gold evidences, (2) using concatenated top-5 BM25 evidences, and (3) using top-5 monoBERT evidences. These experiments aim to assess the impact of multitask pretraining on the classification task and also how classification accuracy depends on the quality of the retrieved evidence.

4 Results

Figure 3 shows our retriever's evaluation results. The left plot (mAP vs top-k) indicates that monoBERT consistently achieves higher mAP scores than the baseline BM25,indicating that is able to rak the gold evidences higher. The middle plot shows that monoBERT also has higher average recall, retrieving more relevant evidences. The right plot confirms monoBERT's overall better retrieval performance with higher average F1 scores. Although the improvements are modest

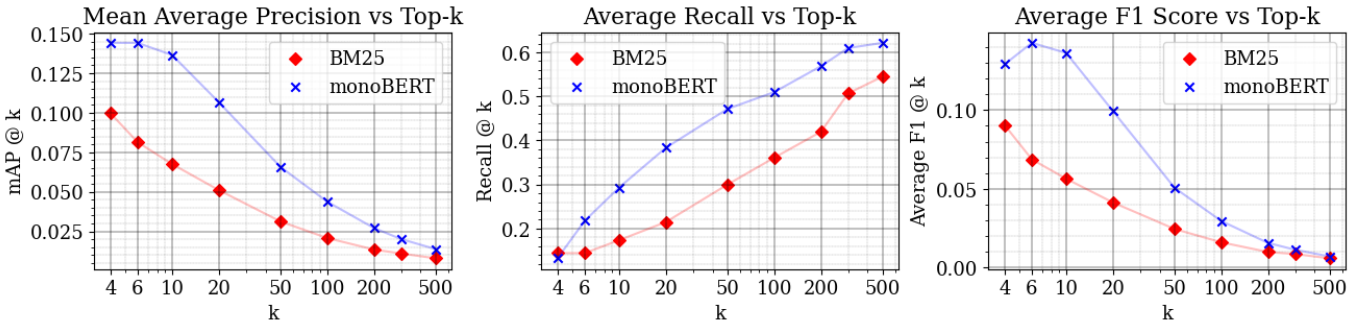

Figure 3: Retriever performance on development set: mAP, Average Recall and Average F1 Score vs top-k.

and not as substantial as we were initially expecting, these results definitively confirm that enhancing BM25 with BERT re-ranking is an effective technique. We hypothesize that the limited performance gain may be due to the small dataset (consisting only of the knowledge source and claim sentences) used for pre-training our custom BERT. Using an open-source transformer model which has been pre-trained on a large corpora will likely lead to more significant improvements, as pre-trained transformers have a better capacity for language understanding compared to BM25, which relies on direct word matching.

Model	**Validation Accuracy**		
	Exp 1	**Exp2**	**Exp3**
Baseline	0.448	0.383	0.396
Finetuned-BERT	0.506	0.422	0.435

Table 2: Classifier performance on development set.

Table 2 presents the results of our experiments on the baseline and fine-tuned BERT models. The fine-tuned BERT model consistently achieves higher accuracy, supporting the merit of our multi-task pre-training strategy. Both models perform best in Experiment 1, where gold evidence passages are used. Additionally, Experiment 3, which uses top-5 monoBERT evidences, yields higher accuracy than Experiment 2 where top-5 BM25 evidences are used. This indicates classification accuracy indeed improves with higher-quality retrieved evidences, as monoBERT provides more relevant passages compared to BM25. Our transformer encoder's block size of 128 required truncation of some evidence passages, introducing up to 2% variability in accuracy. Using a larger block size and including more relevant evidences was too prohibitive, however we would have liked to have done some experiments in this area.

5 Conclusion

In summary, our transformer-based methods of retreival and classification for automated fact-verification have proven to be effective compared to the simpler baseline approach. Despite the limited size of our dataset and small model-scale, our experiments clearly demonstrate that transformer-enhanced retrieval and multi-task pre-training can appreciably improve performance.

However, our work also highlights some major limitations. It is known that transformer models ideally perform best in large data regimes where NLU capabilities begin to emerge. In our case, pre-training on such a small claims dataset and knowledge source posed a risk of overfitting and lack of generalizability. A lack of access to open-source pre-trained transformer models and publicly available fact-verification datasets also hindered further advancements. Furthermore, computational resource constraints prevented us from performing more extensive hyperparameter tuning, and we were also unable to incorporate data augmentation techniques, such as paraphrase generation via back-translation. Additionally, our models relied solely on machine learning techniques, without incorporating any hand-engineered features, external domain knowledge, or rule-based techniques.

While transformer-based language models show great promise, small models pre-trained on limited datasets are insufficient to achieve the necessary level of NLU capability. Our future efforts should focus on pre-training larger models on extensive open corpora, investigating alternative transformer architectures such as transformer decoders and seq2seq models, and exploring the benefits of jointly training on additional tasks related to claim verification. Addressing these areas could lead to more significant performance improvements.

References

Mitchell DeHaven and Stephen Scott. 2023. BEVERS: A general, simple, and performant framework for automatic fact verification. In *Proceedings of the Sixth Fact Extraction and VERification Workshop (FEVER)*, pages 58–65, Dubrovnik, Croatia. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*.

Thomas Diggelmann, Jordan L. Boyd-Graber, Jannis Bulian, Massimiliano Ciaramita, and Markus Leippold. 2020. Climate-fever: A dataset for verification of real-world climate claims. *ArXiv*, abs/2012.00614.

Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. 2022. A survey on automated fact-checking. *Transactions of the Association for Computational Linguistics*, 10:178–206.

Zhen Huang, Shiyi Xu, Minghao Hu, Xinyi Wang, Jinyan Qiu, Yongquan Fu, Yuncai Zhao, Yuxing Peng, and Changjian Wang. 2020. Recent trends in deep learning based open-domain textual question answering systems. *IEEE Access*, 8:94341–94356.

Dan Jurafsky and James H. Martin. 2024. Speech and language processing. 3 edition, chapter 14. Draft.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Zhenghao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2020. Fine-grained fact verification with kernel graph attention network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7342–7351, Online. Association for Computational Linguistics.

Huanhuan Ma, Weizhi Xu, Yifan Wei, Liuji Chen, Liang Wang, Q. Liu, and Shu Wu. 2023. Ex-fever: A dataset for multi-hop explainable fact verification. *ArXiv*, abs/2310.09754.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *ArXiv*, abs/1901.04085.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. 2021. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2523–2544, Online. Association for Computational Linguistics.

Stephen Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. 1995. Okapi at trec-3. In *Overview of the Third Text REtrieval Conference (TREC-3)*, pages 109–126. Gaithersburg, MD: NIST.

Amir Soleimani, Christof Monz, and Marcel Worring. 2020. Bert for evidence retrieval and claim verification. In *Advances in Information Retrieval*, pages 359–366, Cham. Springer International Publishing.

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.

Andrew Yates, Rodrigo Nogueira, and Jimmy Lin. 2021. Pretrained transformers for text ranking: BERT and beyond. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Tutorials*, pages 1–4, Online. Association for Computational Linguistics.

Qin Zhang, Shangsi Chen, Dongkuan Xu, Qingqing Cao, Xiaojun Chen, Trevor Cohn, and Meng Fang. 2023. A survey for efficient open domain question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14447–14465, Toronto, Canada. Association for Computational Linguistics.