COMP90042 Natural Language Processing Assignment 2

Teammate name
Student ID:
The University of Melbourne
`*******@student.unimelb.edu.au`

Kangyu Zhu
Student ID: [redacted]
The University of Melbourne
`kanzhu@student.unimelb.edu.au`

Teammate name
Student ID:
The University of Melbourne
`*******@student.unimelb.edu.au`

1 Abstract

Automated claim verification requires both accurate evidence retrieval and semantic reasoning over large-scale corpora. Here, we develop a four-stage modular pipeline to verify climate-related claims. We integrate symbolic shortlisting based upon lemmatised noun phrases using n-gram indices together with cross-encoders based upon BERT for semantic scoring and final label classification.

On Climate-FEVER dataset, our system outperforms a TF-IDF + Transformer baseline in F1-score and harmonic mean while having excellent accuracy. The pipeline is still light-weight and runs well on free-tier hardware. The system also offers modularity and interpretability for further extension.

2 Introduction & Background

In today's fast-paced information age, spread of misinformation is a significant threat to people's knowledge. Especially in sensitive areas such as climate science. For example, there are some individuals claiming: "The Earth's climate has always changed" (WWF, 2023). However, this statement is often used to dismiss the impact of human activities on current climate change. Scientific evidence shows that human activities accelerate the process. Therefore, automated claim verification is a scalable solution here: for a given factual statement, the system identifies verified evidence and predicts support, refutes, or unverifiability for the statement. This is a core requirement for upholding scientific honesty in society's conversation.

Even with advances in natural language understanding, effective claim verification is still difficult. BERT-like large language models are good at semantic reasoning but computationally expensive (Choi et al., 2021). Symbolic approaches like TF-IDF are fast but brittle when dealt with lexical variation and paraphrasing, because they lack semantic understanding (Algocademy).

To fill this gap, we introduce a hybrid pipeline combining symbolic filtering with deep neural encoding. The system operates well within Google Colab's free tier's memory and runtime limits, making it an ideal low-resource system.

We present the rationale for our design, implementation, and experimental results for our claim verification pipeline in the remaining paper.

3 Data Analysis

3.1 Evidence Corpus

Our evidence collection contains **1 208 827** passages, making it a large and diverse retrieval pool. Figure 4 summarises the passage–length distribution.

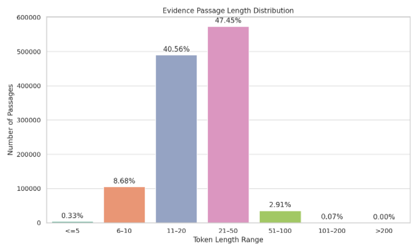

Figure 1: Evidence-passage length distribution.

over 80% of passages fall within the 11–50 word range. This length range offers a good balance between semantic richness and retrieval efficiency. Very short passages ($\leq$ 5words) lack context and may cause false positives in keyword-based methods. However, extremely long passages are difficult to encode effectively without truncation, especially under a 512-token limit.

This skewed length distribution motivates the use of length-aware retrieval heuristics in our pipeline. By filtering out extremely short or long candidates, we improve both relevance and processing efficiency in later stages.

3.2 Claim Label Distribution

The training split comprises **1 228** labeled claims with four possible labels. Figure 2 reveals a clear imbalance: `SUPPORTS` and `NOT_ENOUGH_INFO` together exceed two-thirds of the data, whereas `DISPUTED` accounts for only one claim in ten.

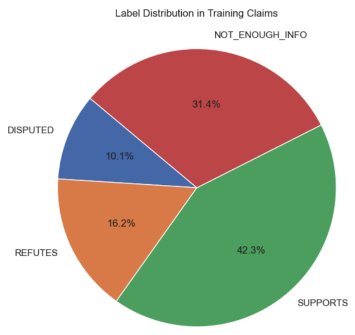

Figure 2: Label distribution in the training set (`train-claims`).

The label distribution in our dataset is skewed. This creates challenges for downstream classification. In particular, the `DISPUTED` class is underrepresented. As a result, the model may struggle to identify ambiguous or borderline claims.

To address this, we use **stratified sampling** in Stage 4 during training. Each mini-batch reflects the overall label distribution. This helps prevent the model from seeing too many examples from dominant classes.

we reduce imbalance effects by using label_weight and sampling techniques in later sections.

These design choices work together to make the system more robust under imbalanced conditions.

4 Workflow

Figure 3 visualises a four–stage pipeline. Each stage is designed to be modular. This allows independent ablation and is easy to replace individual components.

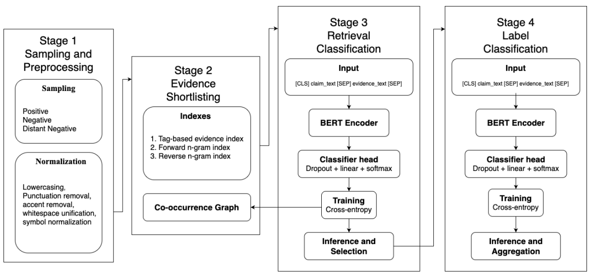

Figure 3: Workflow

4.1 Preprocessing and Shortlisting

To support efficient and accurate claim verification, we designed a two-stage preprocessing and shortlisting pipeline that transforms raw claim-evidence data into structured, high-recall candidate sets. This symbolic pipeline serves as a crucial front-end to downstream deep learning modules, enabling scalable retrieval without sacrificing relevance.

4.1.1 Stage 1: Sampling and Preprocessing

Three types of Sampling. we first construct training examples by generating **positive**, **negative**, and **distant negative** claim–evidence pairs. Positive examples are sourced from manually annotated gold-standard pairs. Negative examples are randomly sampled from the evidence corpus and do not support or refute the claim. To improve the model's discriminative power, we introduce **distant negatives**—passages that share lexical or topical similarity with the claim but are not labeled as evidence. By incorporating contrastive examples across a spectrum of semantic distances, we enhance generalization and robustness in ambiguous or under-specified scenarios.

Claim ID	Claim Text
claim-1937	Not only is there no scientific evidence that ...
claim-1937	Not only is there no scientific evidence that ...
claim-1937	Not only is there no scientific evidence that ...

Table 1: Claim-Evidence Pair sample

Evidence ID	Evidence Text	Related
evidence-442946	At very high concentrations ...	1
evidence-963978	The historical Technology Review ...	0
evidence-787821	Vasudeva, the character from ...	0

Table 2: Evidence Matching for Above Claims

Normalisation. Once sampled, we perform the same text normalisation operations for claims and evidence. These include lowercasing, stripping off punctuation and accents, unification of whitespace, and standardising symbols. We further substitute domain-specialist vocabulary, e.g., "CFC" for "chlorofluorocarbons" and "°C" for "degree Celsius." Next, we employ spaCy for token lemmatisation and removing stop words. This eliminates vocabulary noise and preprocesses the text for retrieval and indexing.

4.1.2 Stage 2: Evidence Shortlisting

- **Two indexes.** The core mechanism for our shortlisting is implemented in the `EvidenceProcessor` module. Two symbolic retrieval indices are built: a **noun-based evidence index** (Thorne et al., 2018) and an **n-gram-based index** (Manber et al., 1997). For the noun index, we extract both single lemmatized nouns and compound noun phrases from each evidence passage (e.g., "carbon emissions", "greenhouse gas", "New York City", "climate change") and map them to the corresponding evidence IDs. This creates a reverse lookup dictionary (`evidence_by_noun.json`) that supports fast retrieval given a claim's noun set.

Noun Phrase	Evidence IDs (Partial)
john	evidence-0, . . . (11667)
john lawes	evidence-0 (1)
bennet	evidence-0, . . . (20)
bennet lawes	evidence-0 (1)

Table 3: Noun-to-Evidence Mapping (Truncated)

  In parallel, we construct an **n-gram** index, both forward and reversed, that captures statistically frequent phrases—4-grams up to 8-grams—that appear across the corpus. These n-grams offer surface-form evidence features that complement noun-based retrieval and allow broader coverage in cases where claims and evidence do not share exact noun matches. Both the noun index and n-gram index are serialized into structured JSON files and used in retrieval.

- **Shortlisting.** During inference, each claim is passed through the same linguistic pipeline to extract base-level concepts, specifically its nouns and proper nouns. These are normalised (via lemmatization and lowercasing) to generate a set of keyword tags. In parallel, character-level prefix and suffix n-grams (of lengths 4 to 8) are extracted for robust partial matching.

  Both tag and n-gram sets are used to independently query three inverted indices: a tag-based evidence index, a forward n-gram index, and a reverse n-gram index. The retrieved evidence IDs are scored using an *inverse document frequency (IDF)*-like scheme:

$$\text{Score}_{evidence} = \sum_{\text{matched terms}} \log_{10}\left(\frac{N}{df}\right) \quad (1)$$

  where N is the total number of distinct evidences across all indices, and df is the number of evidences matched by a particular tag or n-gram. This scoring prioritizes rare but specific matches, discouraging overly frequent, generic tokens.

  The final candidate list for each claim is computed by aggregating scores from all sources, sorting them in descending order, and selecting the top $k = 500$ evidences. This truncated shortlist balances recall and computational cost, and is passed to the downstream retrieval classifier for reranking and final decision.

By eliminating over **99%** of the evidence corpus while keeping a high proportion of gold-standard references, our system enables efficient downstream BERT-based encoding and classification within limited resource environments.

4.2 Stage 3: Retrieval Classification

This stage ranks the evidence candidates shortlisted in Stage 2 by computing the semantic similarity between claim and evidence pairs. It uses a fine-tuned BERT cross-encoder to make predictions (Soleimani et al., 2020).

Input pairs: Each claim and its shortlisted evidence passages are packed in the format: `[CLS] claim_text [SEP] evidence_text [SEP]`, truncated to a maximum of 512 tokens.

Encoder: A `bert-base-uncased` model encodes the concatenated input. Only the `[CLS]` token representation is used for classification.

$$h_{\text{CLS}} = \text{BERT}_{\text{encoder}}([\text{CLS}]\ \text{Claim}\ [\text{SEP}]\ \text{Evidence}\ [\text{SEP}])$$

Classifier head: The pooled output passes through a dropout layer ($p = 0.1$), then a linear layer with output dimension 2, followed by softmax to obtain the final class probabilities. Labels: `0 = Not related`, `1 = Related`.

Training: The classifier is trained with binary cross-entropy loss, using AdamW optimizer and linear learning rate decay. The binary cross-entropy loss $\mathcal{L}_{\text{BCE}}$ is defined as:

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(p_i) + (1-y_i)\log(1-p_i)]$$

Training pairs are constructed with a balanced mix of positive and negative examples based on annotated training data.

Selection: During inference, all evidence candidates for a given claim are scored using the model. The top-k evidences with the highest predicted probability of `Related` (class 1) are retained (typically $k = 5$).

This retrieval classifier ensures that only the most semantically relevant evidences are passed to the final classification stage.

4.3 Stage 4: Label Classification

After retrieving the top-k evidence passages for each claim, we apply a second BERT-based classifier (Soleimani et al., 2020) to predict the final veracity label of the claim.

Architecture: We reuse the same `bert-base-uncased` encoder architecture as Stage 3. A new classification head is attached: Dropout $(p = 0.1) \rightarrow$ Linear $(\text{out\_dim} = 3) \rightarrow$ Softmax:

$$p_{\text{label}} = \text{softmax}(\text{Linear}(\text{Dropout}(h_{\text{CLS}})))$$

Dropout is exposed as a tunable hyperparameter, typically set to 0.1 in our experiments.

Labels: The model performs a 4-way classification with the following labels:

```
0 = REFUTES
1 = NOT_ENOUGH_INFO
2 = SUPPORTS
3 = DISPUTED
```

Input: For each claim and its top-k evidence(s), we construct the input in the format: `[CLS] claim_text [SEP] evidence_text [SEP]`.

Training: The classifier is trained using cross-entropy loss on manually labelled claim-evidence pairs. The cross-entropy loss $\mathcal{L}_{\text{CE}}$ is defined as:

$$\mathcal{L}_{\text{CE}} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{C} y_{ij} \log(p_{ij})$$

To mitigate label imbalance, we apply class weighting based on training set frequency, and optionally use label smoothing to reduce overconfidence. AdamW optimizer and early stopping are used to avoid overfitting.

Inference and Aggregation: For each claim, the classifier predicts a label for each evidence passage in the top-k set. The final verdict is obtained by majority voting or by selecting the label with the highest confidence score among all predictions.

End-to-end behaviour. Each claim is processed through all four stages: preprocessing (Stage 1), evidence shortlisting (Stage 2), retrieval classification (Stage 3), and final label classification (Stage 4). Early symbolic filtering greatly reduces the number of claim–evidence pairs. The system only makes $k \times m$ BERT calls, where m is the number of shortlisted candidates and k is the number kept after scoring. This keeps the pipeline efficient, even on limited resources such as free-tier Colab, while preserving high recall and accuracy. The modular structure allows easy ablation and independent tuning of each stage, balancing precision, scalability, and interpretability.

5 Experiments and Evaluation

Evaluation Metrics

To assess the performance, we use three metrics: **Accuracy**, **F-score**, and their **Harmonic Mean**. Accuracy measures the overall correctness but can be misleading on imbalanced data. F-score, the harmonic mean of precision and recall, balances false positives and negatives. We also report the harmonic mean of accuracy and F-score.

Together, these metrics provide a balanced view of model performance.

Main Result

We first implemented a baseline model using TF-IDF to extract keywords from the claim text and a custom Transformer architecture is used to perform the classification task. As shown in Table 4 and Table 5 , the accuracy on the validation set appears to be satisfying; however, its low F-score and harmonic mean indicate that it is biased toward the majority class, struggles to correctly identify minority classes, and fails to generalise.

To improve performance, we developed this Shortlisting + BERT model. It combines symbolic filtering with semantic encoding. The model scores 0.602 Accuracy, 0.620 F-score and 0.612 Harmonic Mean. This shows that it performs fairly across different classes.

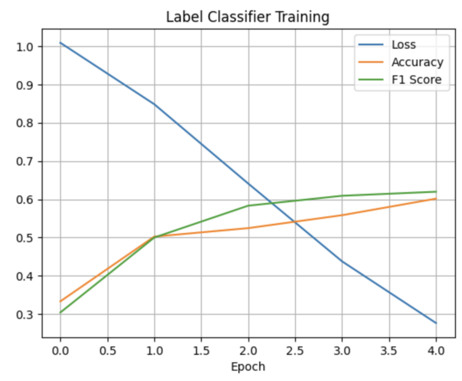

Figure 4: Evidence-passage length distribution.

These results highlight the importance of an excellent evidence retrieval step. Our enhanced pipeline reduces overfitting to common patterns. It also improves the model's ability to detect a wide range of complex claims.

Model	Accuracy	F-score
TF-IDF + Transformer	0.537	0.067
Shortlisting + BERT	**0.602**	**0.620**

Table 4: Classification accuracy on the development set.

Model	Harmonic Mean
TF-IDF + Transformer	0.119
Shortlisting + BERT	**0.612**

Table 5: Retrieval performance on the development set. F-score and harmonic mean reflect macro-level performance across classes.

6 Discussion and Conclusion

6.1 Key Findings

The baseline TF-IDF + Transformer model achieved 0.537 accuracy on the development set but only 0.067 F1-score. This large gap shows it over-predicts the majority class and struggles with less frequent cases.In contrast, our Shortlisting + BERT model scored 0.602 accuracy and 0.620 F-score. Our model excels in both accuracy and f-score compared to the baseline model.

These results highlight that accuracy alone is not a reliable metric in imbalanced datasets. A high accuracy may mask poor generalisation.

Aligned with the original noun-to-evidence mapping, our `EvidenceProcessor` goes further. It adds compound noun extraction, n-gram indexing, and POS-aware retrieval. This results in a flexible and extensible symbolic retrieval module.

Error analysis shows failures mostly occur on claims with rare domain-specific terms (e.g., "albedo feedback"), which often have no matching n-gram in the evidence.

In summary, combining symbolic and neural methods not only improves performance metrics but also delivers more robust and realistic claim verification.

6.2 Limitation

Fixed top-k truncation: During retrieval, only the top-k evidence passages are retained. This may lead to the exclusion of slightly lower-ranked but still relevant evidence, affecting final classification.

Domain coverage: The evidence corpus is static; new climate publications are ignored.

Hardware constraints: We restrict models to fit on free-tier Colab (<12 GB). Larger encoders like RoBERTa-large are therefore excluded.

Label imbalance: Minority classes remain under-represented despite weighting; synthetic augmentation may help.

6.3 Future Work

Multi-hop aggregation: We plan to combine several weakly relevant passages to build stronger support or refutation for complex claims.

Hybrid retrieval: We will explore sparse-dense hybrid retrieval methods. These can help retrieve semantically relevant evidence even when there is little lexical overlap.

Data augmentation: We will experiment with data augmentation techniques. These can increase training diversity and improve model generalisation.

Long and cross-document reasoning: We will extend our approach to handle long documents and multiple sources. This will support claims that require validation across larger or multi-source corpora.

6.4 Final Sentence

In conclusion, we presented a modular pipeline for automated claim verification. the system performs well even in resource-constrained environments. Our findings highlight that retrieval should not be treated as a simple pre-processing step. Instead, it plays a central role in the verification process. Overall, our results show that a carefully designed symbolic layer can significantly enhance the performance of deep learning models in complex reasoning tasks.

References

Algocademy. Understanding tf-idf: The key to smarter information retrieval.

Hyunjin Choi, Judong Kim, Seongho Joe, and Youngjune Gwon. 2021. Evaluation of bert and albert sentence embedding performance on downstream nlp tasks. *arXiv preprint arXiv:2101.10642*.

Udi Manber, Tim Bemers-Lee, George Forman, and Frank W. Tompa. 1997. Performance and scalability of a large-scale n-gram based information retrieval system. *Journal of Digital Information*, 1(4).

Amir Soleimani, Christof Monz, and Marcel Worring. 2020. BERT for Evidence Retrieval and Claim Verification. In *Advances in Information Retrieval: 42nd European Conference on Information Retrieval (ECIR 2020)*, volume 12036 of *Lecture Notes in Computer Science*, pages 359–366. Springer.

James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. 2018. The fact extraction and verification (fever) shared task. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 1–9, Melbourne, Australia. Association for Computational Linguistics.

WWF. 2023. 10 myths about climate change. Accessed: 2025-05-21.

7 Team Contributions

7.1 Zhuo Chen

Zhuo Chen was primarily responsible for designing and implementing the evidence retrieval pipeline. This includes constructing the noun-based and n-gram-based indices, designing the IDF-style scoring system, and building the noun co-occurrence graph for semantic expansion. He also implemented the shortlisting module that reduces the search space from the entire evidence corpus to a manageable set of high-recall candidates. In addition, he contributed to the writing of Section 4.1 in the report and supported experimental setup and debugging.

7.2 Kangyu Zhu

Kangyu Zhu focused on the development and training of the BERT-based retrieval classifier and label classifier. He fine-tuned the cross-encoder model, handled training data generation including positive, negative, and distant negative examples, and optimised hyperparameters. Kangyu Zhu also led the design of the evaluation framework, including the accuracy, F-score, and harmonic mean metrics, and conducted ablation studies. He authored most of Section 5 (Experiments and Evaluation), including result interpretation and metric discussion.

7.3 ZiXuan He

Zixuan He took charge of integrating all components into a coherent end-to-end pipeline and ensuring smooth data flow between stages. He was also responsible for drafting the overall report structure and writing the Introduction, Discussion and Conclusion sections. Additionally, Zixuan He conducted qualitative error analysis and summarized the project's key findings, limitations, implications, and future work. He also contributed to formatting and proofreading the report.