Fraunhofer SIT at CheckThat! 2025: Multi-Instance Evidence Pooling for Numerical Claim Verification

Notebook for the CheckThat! Lab at CLEF 2025

André Runewicz[1,*], Paul Moritz Ranly[1], Inna Vogel[2] and Martin Steinebach[1]

[1]*Fraunhofer Institute for Secure Information Technology SIT | ATHENE - National Research Center for Applied Cybersecurity, Rheinstrasse 75, 64295 Darmstadt, Germany, https://www.sit.fraunhofer.de/*

[2]*Advisori FTC GmbH, Kaiserstrasse 44, 60329 Frankfurt am Main, Germany, https://advisori.de/*

Abstract

The growing spread of misinformation, particularly on social media, has increased the demand for scalable and accurate automated fact-checking systems. This paper presents our approach to Task 3 of the CheckThat! 2025 lab, which focuses on the verification of numerical claims. We propose a three-stage architecture comprising (i) semantic evidence retrieval using dense bi-encoder representations stored in a FAISS index, (ii) contrastive re-ranking with a fine-tuned cross-encoder leveraging weak supervision from gold evidences, and (iii) claim classification using multi-instance learning (MIL) with various evidence pooling strategies. Experiments on the QuanTemp dataset demonstrate that attention and LogSumExp pooling outperform standard concatenation methods, with our best model achieving a macro-F_1 score of 0.5213 on the test set. Additionally, ablation studies confirm the effectiveness of contrastive re-ranking and the practical advantages of dense retrieval, which achieves both higher accuracy and significantly faster retrieval than traditional BM25.

Keywords

Fact-checking, Claim verification, Misinformation detection, Numerical claims, Multi-instance learning, Cross-encoder re-ranking, Contrastive learning, Dense retrieval, Natural language processing

1. Introduction

Misinformation is one of today's greatest challenges in the digital world. Whether they are distributed intentionally or not, they can have a large influence on our opinions and spread rapidly [1]. Since many adults refer to social media platforms when consuming news [2], the risk of further dissemination of disinformation is more present than ever. Conventional fact-checking organizations like *PolitiFact*[1] or *Snopes*[2] struggle to keep up with the amount of information worth checking, as manual fact checking is both time-consuming and tedious [3]. To address these concerns and automatically detect misinformation, Nakov et al. [4] presented the full claim verification pipeline for the CheckThat! 2022 lab. The pipeline consists of four main tasks, namely *check-worthiness estimation*, *verified claim retrieval*, *evidence retrieval* and *claim verification*.

The CheckThat! lab has been addressing different parts of the claim verification pipeline in multiple iterations of its appearance, for instance in its releases in 2018 (check-worthiness detection and claim verification), 2020 (all subtasks) and 2024 (check-worthiness detection) [5, 6, 7]. In the 2025 edition of the CheckThat! lab, task 3 addresses the last step of the pipeline — claim verification — with a focus on numerical claims [8]. Numerical claims are here defined as claims that require validation of quantitative or temporal statements, e.g., a mention of a date or year, or a quantity of participants.

CLEF 2025 Working Notes, 9 – 12 September 2025, Madrid, Spain

*Corresponding author.

andre.runewicz@sit.fraunhofer.de (A. Runewicz); paul.ranly@sit.fraunhofer.de (P. M. Ranly); inna.vogel@advisori.de (I. Vogel); martin.steinebach@sit.fraunhofer.de (M. Steinebach)

0009-0000-2006-5969 (A. Runewicz); 0009-0004-7671-5368 (P. M. Ranly); 0009-0003-9090-5005 (I. Vogel); 0000-0002-0240-0388 (M. Steinebach)

[1] https://www.politifact.com/

[2] https://www.snopes.com/

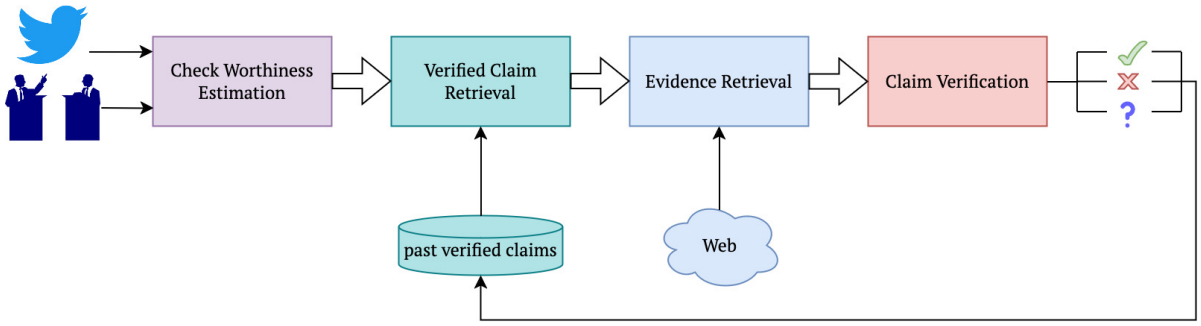

Figure 1: Full verification pipeline including all subtasks as described by Nakov et al. [4].

In this paper, we present our approach to the third task, which follows a three-stage architecture: (i) initial evidence candidate retrieval using dense vector similarity from a FAISS index, (ii) fine-grained re-ranking through a contrastively trained cross-encoder, and (iii) final claim classification via a MIL framework based on `roberta-large-mnli`. Our method emphasizes semantic retrieval over traditional surface-level approaches (e.g., BM25), leverages weakly supervised contrastive signals from gold evidences, and avoids noisy input concatenation by aggregating evidence representations through pooling mechanisms. This modular setup allows the system to flexibly combine retrieval and reasoning, while remaining scalable for real-world application.

Our main contributions are as follows:

- We propose a modular pipeline for claim verification that integrates dense retrieval, contrastive re-ranking, and multi-instance classification for numerical claims.
- We introduce a contrastive training setup using weak supervision from summarized gold evidences to improve re-ranking quality without relying on manual annotation.
- We evaluate multiple evidence aggregation strategies (Max, LogSumExp, Attention), and show that pooling-based MIL improves over standard concatenation approaches.

The paper is structured as follows: Section 2 gives an overview of the fake news detection pipeline as presented in [4] and the problem of the specific task. Section 3 summarizes previously conducted works in this field. In Section 4 we present our methodology, which we then evaluate and analyze in Section 5. Finally, we provide a conclusion and future outlook to our approach in Section 6.

2. Background

This section provides an overview of the fake news detection pipeline (Subsection 2.1) and presents a formal definition of the claim verification task (Subsection 2.2).

2.1. Fake News Detection Pipeline

The process of detecting and evaluating fake news was first formulated by Nakov et al. in 2018 [5] and further refined in the following editions. Since 2020 the pipeline is presented in four distinct stages. We will look at an adaption from the pipeline as shown in Figure 1.

The first task, *check-worthiness estimation*, assesses whether a statement needs further verification, i.e., if it needs to be passed on to a verifying authority. The second task, *verified claim retrieval*, attempts to determine whether the claim has already been fact-checked by searching a database of previously verified claims. If a matching verified claim is found, its label can directly support the classification of the new claim. The third step, *evidence retrieval*, involves gathering additional relevant evidence from external sources, such as web documents or news outlets, which may help in verifying the claim. Finally, the fourth task, *claim verification*, combines the original claim with the retrieved evidences to assess its veracity. This is described in more detail in the following Subsection.

2.2. Problem Formulation

The task of claim verification involves determining the veracity of a claim by consulting a set of evidences retrieved from a large corpus. Formally, given a query (claim) and relevant evidences, the goal is to predict whether the claim is *true*, *false*, or in *contradiction* with the available information. Mathematically, this can be described as follows:

1. Definitions

Let:

- $\mathcal{Q} = \{q_1, q_2, \ldots, q_N\}$ be the set of *queries* (claims).
- $\mathcal{C} = \{e_1, e_2, \ldots, e_M\}$ be the *corpus of evidences*.
- $f : \mathcal{Q} \times \mathcal{C}^k \to \mathcal{Y}$ be the *verification model*, which maps a query and a set of k evidences to a label.
- $\mathcal{Y} = \{\text{True}, \text{False}, \text{Contradiction}\}$ be the label space.

2. Evidence Retrieval Function

Let r be the *retrieval function*, selecting the top-k relevant evidence passages from the corpus for a given query q:

$$r : \mathcal{Q} \to \mathcal{C}^k \tag{1}$$

This can be formalized as:

$$r(q) = \arg \max_{E \subset \mathcal{C}, |E|=k} \sum_{e \in E} \text{rel}(q, e) \tag{2}$$

where $\text{rel}(q, e)$ is a relevance scoring function.

3. Classification Objective

Given a query $q \in \mathcal{Q}$ and its retrieved evidence set $E = r(q)$, the classifier predicts:

$$\hat{y} = f(q, E) \tag{3}$$

where $\hat{y} \in \mathcal{Y}$ is the predicted label.

4. Learning Objective

Given a training data set $\{(q_i, E_i, y_i)\}_{i=1}^{N}$, where y_i is the ground truth label for query q_i with evidence set E_i, the model parameters θ are optimized by minimizing the classification loss:

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \ell(f_\theta(q_i, E_i), y_i) \tag{4}$$

where ℓ is a loss function (cross-entropy).

2.3. Dataset

The relevant dataset for the claim verification task in english is QuanTemp [9], a multi-domain dataset utilizing real-world data and focusing on numerical claims. While past iterations of the task comprised of political (mainly PolitiFact) or social media data (from X, formerly Twitter) [5, 6], QuanTemp consists of multiple domains as thus has more varied appliances. The detailed distribution of the dataset as provided by the authors is given in Table 1.

Table 1
Distribution of claims in QuanTemp (adapted from Venktesh et al. [9]). Note: While the official test set size is 2,495, we were actually provided with 3,656 instances.

Split	True	False	Conflicting	Total
Train	1,824	5,770	2,341	9,935
Dev	617	1,795	672	3,084
Test	474	1,423	598	2,495

As can be seen, the dataset is imbalanced and consists mostly of false claims. It can be argued that the sources of the data, which comes mostly from PolitiFact and Snopes, effects the false classifications, as fact-checking organizations mainly deal with questionable or incorrect claims which need further verification.

3. Related Work

Claim verification as a standalone task was first presented in 2018 and the next two editions. It was then reintroduced in the 2023 `CheckThat!` lab. It has been held in both english and arabic language, but it generally has not had as many participants as other tasks in the domain of the full claim verification pipeline. In most cases, the participants retrieved external information to be able to satisfactory classify the given claims [5]. Methods used for classification include more traditional approaches like SVMs, random forest classifiers or logistic regression [10], but also neural architecures like CNNs [11]. Ghanem et al. [12] proposed a three-stage verification approach, consisting of evidence retrievel, evidence ranking and textual entailment. For the 2023 task, which was held in arabic, Touahri and Mazroui [13] implemented linguistic features that focus on the alignment between claims and externally extracted texts.

Apart from challenges in the `CheckThat!` lab, approaches utilized Transformer models like BERT [14] or other language model ensembles [15] for claim verification. More recently, architectures based on graph networks [16] and large language models (LLMs) [17] also showed promising results and outperformed many traditional approaches.

Many of the presented approaches were evaluated using FEVER [18], a dataset consisting of over 185,000 generated claims used for claim verification. In 2024, Venktesh et al. [9] introduced a novel real-world dataset, where all claims are sourced from real fact-checks. The data focuses on claims that include numerical claims. Furthermore, they also experimented with claim decomposition to improve claim verification, which showed promising results on the QuanTemp dataset.

4. Methodology

Our submitted pipeline (english only) follows a three-stage architecture: (i) evidence candidate retrieval using dense vectors, (ii) re-ranking using a fine-tuned cross-encoder, and (iii) final claim classification using a large NLI model. In the following, each step is described in detail.

Evidence Retrieval We first pre-computed document embeddings using `sentence-transformers/all-MiniLM-L6-v2`[3] and stored them in a FAISS index for efficient inner-product search. The embeddings were L2-normalized to enable cosine similarity retrieval. At inference time, each claim was encoded into a query embedding, normalized, and used to retrieve the top-100 most similar evidence snippets from the index. The goal was to retrieve not only lexically similar evidence snippets, but also semantically relevant ones that may be phrased differently.

[3] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

Re-ranking with Contrastive Fine-Tuning The initial candidates were re-ranked using `cross-encoder/ms-marco-MiniLM-L-6-v2`[4], fine-tuned in a contrastive setup. To improve re-ranking beyond the vanilla MS MARCO model, we generated weak supervision signals from gold-labeled evidence snippets in the training and validation sets. Further, these snippets were summarized using `LLaMA-3.1-8B`[5] to remove noise and produce cleaner positives. During training, the model was optimized using margin ranking loss[6], where each claim served as an anchor, its corresponding summarized gold evidence as the positive, and the top-100 BM25-retrieved candidates as negatives. This training strategy encouraged the model to assign higher relevance scores to gold summaries by pulling them closer to the claim in the embedding space. Crucially, this step addressed the main bottleneck in the task: retrieval quality.

Claim Classification with Multi Instance Learning The final decision was produced by `FacebookAI/roberta-large-mnli` trained in a *MIL* setup. For every claim $q \in \mathcal{Q}$, we took the $k = 10$ re-ranked evidence passages $E = \{e_1, \ldots, e_k\} \subset \mathcal{C}$, and forwarded each pair (q, e_i) through the encoder, obtaining a class-logit vector $s_i \in \mathbb{R}^C$, with $C = 3$ (True, False, Contradiction). The k snippet-level logit vectors were then *pooled* into a single bag-level vector $S^{(\cdot)}(q, E) \in \mathbb{R}^C$ before applying softmax. Each pooling operator produced a scalar score $S_\ell^{(\cdot)}(q, E)$ for class ℓ. The final prediction was obtained by:

$$\hat{y} = \arg\max_{\ell} S_\ell^{(\cdot)}(q, E) \tag{5}$$

where $(\cdot)$ denotes the choice of pooling operator:

1. **Max pooling** (hard winner-take-all) picks the single most confident snippet per class:

$$S_\ell^{\max}(q, E) = \max_{1 \le i \le k} s_{i,\ell}, \tag{6}$$

   where $s_{i,\ell}$ is the logit of class ℓ.
2. **LogSumExp pooling** is a smooth approximation of the maximum:

$$S_\ell^{\text{lse}}(q, E) = \log\Big(\sum_{i=1}^{k} \exp s_{i,\ell}\Big). \tag{7}$$

3. **Attention pooling** learns a soft relevance weight for every snippet:

$$\alpha_i = \frac{\exp\big(a^\top h_i\big)}{\sum_{j=1}^{k} \exp\big(a^\top h_j\big)}, \tag{8}$$

$$S_\ell^{\text{attn}}(q, E) = \sum_{i=1}^{k} \alpha_i \, s_{i,\ell}, \tag{9}$$

   where $h_i \in \mathbb{R}^d$ is the CLS embedding of (q, e_i) and $a \in \mathbb{R}^d$ is a single learned parameter vector ($d = 1024$ for `roberta-large`).

Max pooling simply takes the highest logit among all snippets for each class and ignores the rest. It is noise-tolerant, but discards any partial support from other evidences. Moreover, it performs poorly when all top-k evidence snippets are irrelevant, as no useful evidence contributes to the prediction.

LogSumExp pooling is a smooth approximation of max and amplifies strong evidence while still giving weaker evidences some influence, which makes it more stable than hard max.

Attention-based pooling, used in our final submission, allows the model to focus on the most informative snippets while ignoring irrelevant ones, without requiring an arbitrary truncation or noisy concatenation of top-k candidates. Furthermore, it is naturally robust to variable evidence quality across claims. In essence, it performs an additional, internal ranking of evidence snippets during classification, letting the model weigh each piece of evidence according to its usefulness for the final decision.

[4] https://huggingface.co/cross-encoder/ms-marco-MiniLM-L6-v2
[5] https://huggingface.co/meta-llama/Llama-3.1-8B
[6] https://pytorch.org/docs/stable/generated/torch.nn.MarginRankingLoss.html

Training Details To ensure comparability with Venktesh et al. [9], all models were trained using only the official data splits provided in the task description/GitLab repository. We did not use any external data sets or knowledge bases. The training and validation splits exactly followed those defined in the official paper. While it might have been possible to significantly boost test performance by externally scraping sources like Google Fact Check[7], we deliberately avoided such shortcuts to stay within the intended constraints of the task.

Training was conducted on an NVIDIA L4 GPU with 22.5 GB VRAM. The model was trained for up to 10 epochs with early stopping (patience = 3), and best validation performance was reached between epochs 5 and 6. We used a batch size of 3 with gradient accumulation steps of 8 (effective batch size of 24). The learning rate was set to 2e-5, with a cosine scheduler and 10% linear warm-up. The model was compiled using torch.compile for performance, and trained using bfloat16 precision via PyTorch AMP. We used a class-weighted cross-entropy loss to handle class imbalance, which led to improved macro-F_1, albeit with a slight decrease in weighted-F_1. For reproducibility, all experiments were run with random seed 42.

5. Results & Analysis

In this section, we present the results of our main method, followed by ablation studies on the test set, and a summary of additional experiments and insights.

Results on Development Set We first evaluated our approach using three different pooling strategies: attention-based, LogSumExp, and max pooling. All experiments used the same retriever (all-MiniLM-L6-v2) and the contrastively fine-tuned cross-encoder described in Section 4. As shown in Table 2, attention-based pooling achieved the best performance, with a macro-F_1 of 0.5937 and a weighted-F_1 of 0.6682. LogSumExp and max pooling performed comparably but slightly below attention pooling. Based on these results, we selected the attention-based pooling model as our final submission.

Table 2
Performance comparison of different pooling strategies (Attention-based, LogSumExp, Max) on the development set using the same retriever (all-MiniLM-L6-v2) and contrastively fine-tuned cross-encoder.

Pooling Method	Macro-F_1	Weighted-F_1
Attention-based	**0.5937**	0.6682
LogSumExp	0.5839	0.6620
Max	0.5852	0.6619

Ablation Studies on Test Set To better understand the impact of individual components, we conducted ablation studies on the test set (see Table 3). Specifically, we varied the retriever (all-MiniLM-L6-v2 vs. BM25) and the re-ranker (fine-tuned vs. vanilla) across the three pooling strategies. Several key observations emerged:

- **Retriever quality matters:** The bi-encoder (MiniLM) consistently outperformed BM25 across all pooling methods. Furthermore, retrieving the top-100 initial candidates using pre-computed MiniLM embeddings with a FAISS index was approximately six times faster than BM25 retrieval, making it significantly more practical for large-scale applications.
- **Contrastive fine-tuning helps:** In all configurations using MiniLM, the fine-tuned re-ranker outperformed the vanilla version, confirming the effectiveness of our contrastive training strategy with summarized gold evidences. At this point, it is worth noting that we used all top-100 retrieved evidence snippets as negatives during training. This simplification is theoretically suboptimal, since some of these snippets may actually be relevant, and treating them as negatives could

[7] https://toolbox.google.com/factcheck

push useful evidence away from the claim in the embedding space. Due to the absence of labels (i.e., which specific snippets are truly supportive for a given claim), we adhered to this heuristic approach.
- **Pooling strategy robustness:** LogSumExp pooling achieved the highest macro-F_1 (0.5213), suggesting that it generalized slightly better to the test set than attention or max pooling.

Interestingly, when using BM25 for retrieval, the vanilla re-ranker occasionally performed better than its fine-tuned counterpart, likely due to BM25 returning noisier candidates that disrupted the fine-tuned model's learned ranking behavior. Overall, the results affirm the benefits of our modular architecture, particularly the fine-tuned re-ranking step and the use of attention-based or LogSumExp pooling in a MIL framework.

Table 3
Ablation results on the test set showing the effect of different retrievers (all-MiniLM-L6-v2 vs. BM25) and re-rankers (fine-tuned vs. vanilla) across three pooling strategies. * indicates the submitted run.

Pooling method	Retriever	Re-ranker	Macro-F_1	Micro-F_1	Weighted-F_1	F_1-True	F_1-False	F_1-Conflicting
Attention-based	all-MiniLM-L6-v2	Fine-tuned	**0.5100***	0.6157	0.6198	0.4087	0.7625	0.3589
	all-MiniLM-L6-v2	Vanilla	0.5009	0.5749	0.5972	0.4292	0.7217	0.3518
	BM25	Fine-tuned	0.4906	0.6127	0.6089	0.3511	0.7636	0.3571
	BM25	Vanilla	0.4966	0.5766	0.5959	0.4131	0.7248	0.3518
LogSumExp	all-MiniLM-L6-v2	Fine-tuned	**0.5213**	0.6275	0.6280	0.4489	0.7657	0.3495
	all-MiniLM-L6-v2	Vanilla	0.5048	0.6089	0.6139	0.4246	0.7551	0.3348
	BM25	Fine-tuned	0.5057	0.6294	0.6203	0.4081	0.7689	0.3402
	BM25	Vanilla	0.5079	0.6182	0.6189	0.4089	0.7631	0.3517
Max	all-MiniLM-L6-v2	Fine-tuned	**0.5116**	0.6204	0.6210	0.3594	0.7649	0.4104
	all-MiniLM-L6-v2	Vanilla	0.5112	0.6059	0.6159	0.3868	0.7530	0.3938
	BM25	Fine-tuned	0.5038	0.6269	0.6208	0.3333	0.7750	0.4031
	BM25	Vanilla	0.5037	0.6080	0.6128	0.3591	0.7559	0.3962

When inspecting the classwise F_1 scores (Table 3) together with the confusion matrices in Figure 2, three clear trends appear:

First, the False label is by far the easiest to recognize. All three pooling strategies achieve an F_1 between 0.75 and 0.78, which is driven by the large number of correct False predictions ($\approx$ 1,700 instances in every matrix) and relatively few confusions with the other classes. Because this class accounts for roughly two-thirds of the test set, its high F_1 score drives the micro- and weighted-F_1 (macro-F_1 only proportional).

Second, the True label benefits most from the smoother LogSumExp pooling. Compared with attention pooling, LogSumExp leaves the number of True→False errors virtually unchanged (252 vs. 254) but substantially increases correct True→True hits (264 → 338) and cuts True→Conflicting confusions (201 → 125), raising the class-wise F_1 from 0.4087 to 0.4489. In contrast, Max pooling sacrifices True accuracy (F_1=0.3594) by sending many true claims to the Conflicting bucket (260 instances). This indicates that its hard selection amplifies noise in borderline evidence.

Third, the Conflicting label remains the hardest case overall. Even the best setting (Max pooling, F_1=0.4104) still misclassifies roughly half of the conflicting claims, mostly as False. This corroborates the qualitative observation that distinguishing genuinely contradictory evidence from merely missing or weak evidence is challenging. The models often fall back to the majority False decision when evidence is ambiguous.

In general, LogSumExp offers the best balance between True and False, whereas Max pooling excels at recognizing Conflicting cases, and attention pooling sits in-between. These findings highlight a trade-off: improving minority classes (True, Conflicting) comes at a slight cost to the False performance, and vice-versa.

Further Experiments and Findings We explored a number of additional modeling strategies:

- **Claim decomposition:** Incorporating decomposed claims (as proposed in Venktesh et al. [9]) did not improve performance. In fact, using the claim-only method yielded slightly better results.

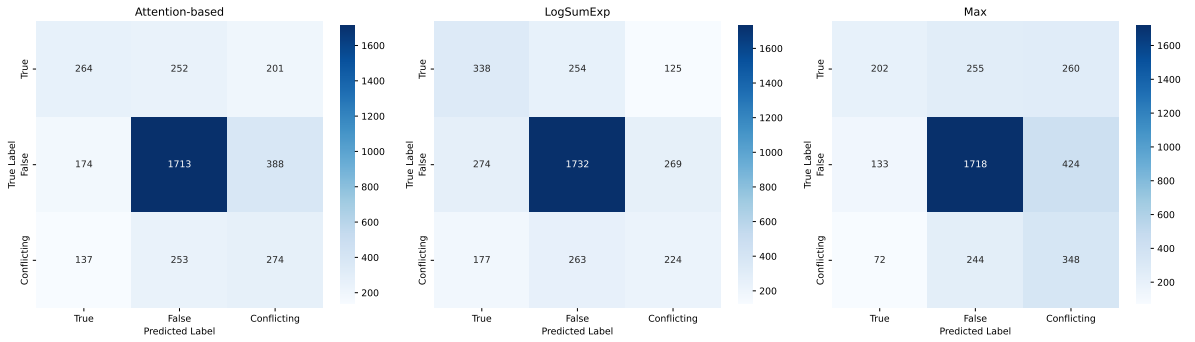

Figure 2: Confusion matrices for the three pooling methods on test set (all using all-MiniLM-L6-v2 as retriever and fine-tuned cross-encoder as re-ranker).

- **Loss functions:** A class-weighted cross-entropy loss proved effective in addressing class imbalance, boosting macro-F_1 by 3–4% at the expense of a minor drop in weighted-F_1 (approximately 2%). Focal loss was also tested but underperformed slightly in comparison.
- **Evidence encoding strategies:** Prior to adopting the MIL setup, we trained a classifier using concatenated top-k evidence snippets as input. However, this approach performed worse than all three MIL-based pooling strategies, with a best macro-F_1 of 0.5850 and weighted-F_1 of 0.6557.
- **Alternative classifiers:** We explored several alternative cross-encoder architectures from the `cross-encoder` library[8], including `ms-marco-MiniLM-L-6-v2`, `stsb-roberta-base`, and `nli-deberta-v3-base`. However, none of these models surpassed the performance of `FacebookAI/roberta-large-mnli`. Moreover, we experimented with using a Graph Attention Network (GAT) for the final classification step. While promising in theory, the GAT-based model underperformed compared to the `roberta-large-mnli` classifier, achieving macro-F_1 and weighted-F_1 scores approximately 4% lower (best run: macro-F_1 = 0.5394, weighted-F_1 = 0.6329).

6. Conclusion and Future Work

This paper presented a three-stage architecture for numerical claim verification in the `CheckThat!` 2025 lab. We first retrieve evidence passages with a dense bi-encoder stored in a FAISS index, then refine the candidate list with a contrastively fine-tuned cross-encoder, and finally classify each claim in a MIL framework based on `roberta-large-mnli` that aggregates snippet-level logits through alternative pooling operators. Evaluated on the QuanTemp dataset, the complete system reaches a macro-F_1 of 0.5213 and a weighted-F_1 of 0.6280 with LogSumExp pooling, which outperforms concatenation-based baselines and demonstrates that pooling-based MIL offers a viable alternative to simplistic evidence concatenation. Ablation studies confirm that dense retrieval offers both higher accuracy and a six-fold speed-up over BM25, while the contrastive fine-tuning of the re-ranker consistently enhances ranking quality.

Despite these gains, the overall macro-F_1 is still capped by the recall of the retrieval stage, which remains the main bottleneck of this task. Future work could therefore focus on retrieval-centric improvements, including late-interaction models such as ColBERT that combine token-level matching with efficient vector search. In addition, manually labeling the truly relevant evidence snippets for each claim would allow us to avoid treating helpful passages as negatives during contrastive training, providing much cleaner learning signals for the re-ranker.

[8] https://sbert.net/examples/sentence_transformer/applications/retrieve_rerank/README.html#pre-trained-cross-encoders-re-ranker

Declaration on Generative AI

During the preparation of this work, we used ChatGPT-4o to assist with light editing tasks such as grammar correction, writing style improvement, and occasional rephrasing. At no point was any section of the paper fully generated by the tool. All content and ideas originate from us. All outputs from the AI assistant were critically reviewed, revised where necessary, and verified by us. We take full responsibility for the content of this publication and confirm that the use of AI-assisted tools is in accordance with the CEUR-WS policy on AI-assisted writing.

Acknowledgments

This research work was supported by the National Research Center for Applied Cybersecurity ATHENE. ATHENE is funded jointly by the German Federal Ministry of Research, Technology and Space and the Hessian Ministry of Science and Research, Arts and Culture.

References

[1] E. Aïmeur, S. Amri, G. Brassard, Fake news, disinformation and misinformation in social media: a review, Social Network Analysis and Mining 13 (2023) 30.

[2] P. R. Center, Social media and news fact sheet, 2023. URL: https://www.pewresearch.org/journalism/fact-sheet/social-media-and-news-fact-sheet/, accessed: 2025-05-27.

[3] S. Shaar, N. Georgiev, F. Alam, G. D. S. Martino, A. Mohamed, P. Nakov, Assisting the human fact-checkers: Detecting all previously fact-checked claims in a document, arXiv preprint arXiv:2109.07410 (2021).

[4] P. Nakov, G. Da San Martino, F. Alam, S. Shaar, H. Mubarak, N. Babulkov, Overview of the clef-2022 checkthat! lab task 2 on detecting previously fact-checked claims (2022).

[5] P. Nakov, A. Barrón-Cedeno, T. Elsayed, R. Suwaileh, L. Màrquez, W. Zaghouani, P. Atanasova, S. Kyuchukov, G. Da San Martino, Overview of the clef-2018 checkthat! lab on automatic identification and verification of political claims, in: Experimental IR Meets Multilinguality, Multimodality, and Interaction: 9th International Conference of the CLEF Association, CLEF 2018, Avignon, France, September 10-14, 2018, Proceedings 9, Springer, 2018, pp. 372–387.

[6] A. Barrón-Cedeño, T. Elsayed, P. Nakov, G. Da San Martino, M. Hasanain, R. Suwaileh, F. Haouari, N. Babulkov, B. Hamdan, A. Nikolov, et al., Overview of checkthat! 2020: Automatic identification and verification of claims in social media, in: Experimental IR Meets Multilinguality, Multimodality, and Interaction: 11th International Conference of the CLEF Association, CLEF 2020, Thessaloniki, Greece, September 22–25, 2020, Proceedings 11, Springer, 2020, pp. 215–236.

[7] A. Barrón-Cedeño, F. Alam, J. M. Struß, P. Nakov, T. Chakraborty, T. Elsayed, P. Przybyła, T. Caselli, G. Da San Martino, F. Haouari, et al., Overview of the clef-2024 checkthat! lab: check-worthiness, subjectivity, persuasion, roles, authorities, and adversarial robustness, in: International Conference of the Cross-Language Evaluation Forum for European Languages, Springer, 2024, pp. 28–52.

[8] F. Alam, J. M. Struß, T. Chakraborty, S. Dietze, S. Hafid, K. Korre, A. Muti, P. Nakov, F. Ruggeri, S. Schellhammer, V. Setty, M. Sundriyal, K. Todorov, V. Venktesh, Overview of the CLEF-2025 CheckThat! Lab: Subjectivity, fact-checking, claim normalization, and retrieval, in: J. Carrillo-de Albornoz, J. Gonzalo, L. Plaza, A. García Seco de Herrera, J. Mothe, F. Piroi, P. Rosso, D. Spina, G. Faggioli, N. Ferro (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Sixteenth International Conference of the CLEF Association (CLEF 2025), 2025.

[9] V. Venktesh, A. Anand, A. Anand, V. Setty, Quantemp: A real-world open-domain benchmark for fact-checking numerical claims, in: 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2024, Association for Computing Machinery (ACM), 2024, pp. 650–660.

[10] K. Yasser, M. Kutlu, T. Elsayed, bigir at clef 2018: Detection and verification of check-worthy political claims., in: CLEF (Working Notes), 2018.
[11] D. Wang, J. Simonsen, B. Larseny, C. Lioma, The copenhagen team participation in the factuality task of the competition of automatic identification and verification of claims in political debates of the clef-2018 fact checking lab, Cappellato et al.[5] (2020).
[12] B. Ghanem, G. Glavaš, A. Giachanou, S. P. Ponzetto, P. Rosso, F. Rangel, Upv-uma at checkthat! lab: verifying arabic claims using a cross lingual approach, in: CEUR Workshop Proceedings, volume 2380, RWTH Aachen, 2019, pp. 1–10.
[13] I. Touahri, A. Mazroui, Evolutionteam at clef2020-checkthat! lab: Integration of linguistic and sentimental features in a fake news detection approach., in: CLEF (working notes), 2020.
[14] A. Soleimani, C. Monz, M. Worring, Bert for evidence retrieval and claim verification, in: European Conference on Information Retrieval, Springer, 2020, pp. 359–366.
[15] S. Gurrapu, L. Huang, F. A. Batarseh, Exclaim: Explainable neural claim verification using rationalization, in: 2022 IEEE 29th Annual Software Technology Conference (STC), IEEE, 2022, pp. 19–26.
[16] Y. Chen, H. Liu, Y. Liu, R. Yang, H. Yuan, Y. Fu, P. Zhou, Q. Chen, J. Caverlee, I. Li, Graphcheck: Breaking long-term text barriers with extracted knowledge graph-powered fact-checking, arXiv preprint arXiv:2502.16514 (2025).
[17] G. Fenza, D. Furno, V. Loia, P. P. Trotta, Multi-llm agents architecture for claim verification (2025).
[18] J. Thorne, A. Vlachos, C. Christodoulopoulos, A. Mittal, Fever: a large-scale dataset for fact extraction and verification, arXiv preprint arXiv:1803.05355 (2018).