PrompterSquad at CheckThat! 2026: Self-Distilled Hard Negatives and a Hybrid Pairwise–Listwise Ranker for Fact-Checking Numerical Claims

Pham Thai Son[1,*], Nguyen Ho Duy Tri[1]

[1]University of Information Technology, VNU-HCM, Ho Chi Minh City, Vietnam

Abstract

This paper describes the PrompterSquad system submitted to Task 2 of the CLEF 2026 CheckThat! Lab, which addresses test-time scaling for fact-checking of numerical claims. Given a claim, a set of retrieved evidence passages, and multiple candidate reasoning traces produced by a Large Language Model (LLM), the task requires participants to (i) rank the reasoning traces by their utility in leading to the correct verdict, and (ii) predict the final verdict from the top-ranked traces. We frame this as a learning-to-rank problem and fine-tune `microsoft/deberta-v3-base` as a pointwise scorer using a hybrid loss that combines a pairwise margin objective with a listwise ListMLE term. We further apply LoRA-based parameter-efficient adaptation and a self-distilled hard negative mining procedure that begins after the second epoch. On the official English test set, our system achieves an average score of 0.4138 (Macro-F1 = 0.5967, Recall@5 = 0.2308), ranking 9th of 9 teams within a spread of only 0.0148 points. A controlled ablation on the held-out validation half identifies the hybrid loss and hard negative mining as the two training-side components contributing meaningful gains; the inference-time refinements (Monte Carlo dropout averaging and multi-k weighted voting), which we deployed in our submission, do *not* produce a measurable benefit in the ablation and we report this honestly. Through an oracle upper-bound analysis we further uncover two structural properties of the task: (i) 29.4% of validation claims have *no positive coverage*-no trace in their pool has an intermediate verdict matching the gold label-imposing a hard ceiling of $\sim$0.67 Macro-F1 on any verifier, and explaining the unusually narrow 0.015-point spread among the top-9 teams; and (ii) the Conflicting class is *learnably* hard rather than intrinsically hard-an oracle ranker achieves 0.68 F1 on Conflicting, more than three times the best participating system, indicating that the bottleneck is verifier learning rather than data ambiguity. We release our code and trained model checkpoints to support reproducibility.

Keywords

fact-checking, numerical claim verification, learning-to-rank, DeBERTa, LoRA, reasoning trace ranking, test-time scaling, CheckThat! 2026

1. Introduction

Automatically verifying claims with quantitative or temporal expressions is a persistent challenge: the model must jointly retrieve statistics, perform arithmetic, and interpret pragmatic qualifiers. Task 2 of the CLEF 2026 CheckThat! Lab [6, 7], built on QuanTemp [15], frames this as a *test-time scaling* problem [16, 1]: given a fixed pool of LLM-generated reasoning traces and their intermediate verdicts, participants must (i) rank traces by utility for reaching the correct verdict and (ii) aggregate a final verdict from the top-ranked traces. Unlike self-consistency, where a simple majority vote suffices, here the aggregator must operate over reasoning paths, requiring a verifier that is simultaneously a ranker and a verdict classifier.

We describe the **PrompterSquad** system (Codabench: `sonpham8112005`): a parameter-efficient cross-encoder on `microsoft/deberta-v3-base` [10] with LoRA [11]. Our key contributions are:

- A **hybrid pairwise-listwise loss** (MarginRankingLoss $\times 0.7$ + ListMLE [17] $\times 0.3$) - the largest single contributor in ablation ($\Delta = +0.0071$ combined).
- **Self-distilled hard negative mining** from epoch 2: the current model re-scores negatives and oversamples the top-60% hardest - second-largest contributor ($\Delta = +0.0033$).

CLEF 2026: Conference and Labs of the Evaluation Forum, September 21–24, 2026, Jena, Germany

*Corresponding author.

23521361@gm.uit.edu.vn (P. T. Son); trinhd@uit.edu.vn (N. H. D. Tri)

- An **oracle upper-bound analysis** revealing: (i) 29.4% of validation claims have no positive trace, capping Macro-F1 at ∼0.67 and explaining the narrow 0.015-point leaderboard spread; (ii) the Conflicting class is *learnably* hard - oracle achieves 0.68 F1 vs. 0.24 for the best team.

On the official English test set we achieve an average score of **0.4138** (Macro-F1 = 0.5967, Recall@5 = 0.2308), ranking 9th of 9 teams. As our oracle analysis shows, this ranking reflects a near-saturated task with a tight 0.0148-point field rather than a large quality gap: our Macro-F1 (0.5967) is within 0.018 of the top system and within 0.074 of the oracle ceiling (Section 6.5), and we trade a lower Recall@5 for a competitive Macro-F1 (Section 6.1). We also report an *honest negative result*: the two inference-time refinements deployed in our official submission (Monte Carlo dropout averaging and multi-k weighted voting) do *not* improve the validation combined score in a controlled ablation, so we de-emphasise them relative to the original submission title and foreground the load-bearing training-side contributions.

2. Related Work

Fact-checking of numerical claims. Venktesh et al. [15] released QuanTemp, the dataset underlying CheckThat! 2026 Task 2. Earlier editions [4, 5] focused on verdict prediction; the 2026 edition is the first to treat the reasoning trace as an explicit evaluation object. Concurrent work [3, 19] shows that intermediate reasoning steps improve verdict accuracy for complex claims.

Learning-to-rank and hard negatives. Cross-encoder rerankers [14, 9] have become standard in second-stage retrieval, trained with pairwise or listwise objectives such as ListMLE [17]. Our hybrid loss combines both. ANCE [18] popularised periodic hard-negative refresh using the current model, which we adapt to trace ranking.

Parameter-efficient fine-tuning and test-time scaling. LoRA [11] makes multi-run experimentation tractable by injecting low-rank updates into a frozen backbone. Test-time scaling via majority vote [16] or learned verifiers [12] motivates the Task 2 evaluation paradigm.

3. Task and Data

3.1. Task Description and Evaluation

Each instance is a tuple (c, E, T, V): a claim c, retrieved evidence $E=\{e_1, \ldots, e_m\}$, LLM-generated reasoning traces $T=\{t_1, \ldots, t_n\}$, and intermediate verdicts $V=\{v_j\}$ where $v_j \in \{\text{True, False, Conflicting}\}$. The system must output (1) a ranked ordering π of traces and (2) a final verdict $\hat{y}$.

The official leaderboard metric is the average of **Recall@5** (fraction of gold-positive traces in the top 5) and **Macro-F1** (unweighted mean of per-class F1 over the three verdict classes). MRR@5 and per-class F1 are also reported.

3.2. Dataset

We focus on the English track. Table 1 summarises the key statistics. We split the validation set in half: 50% for additional training, 50% held out for evaluation. Two data properties are central to our analysis: (1) **label imbalance** - False dominates at 57.1%, making Conflicting (23.9%) and True (19.0%) the harder classes; (2) **incomplete trace coverage** - 29.4% of validation claims have no trace whose intermediate verdict matches the gold label, imposing a hard ceiling on any verifier.

Table 1
Dataset statistics (English track, after pre-processing).

Quantity	Value
Training claims	2,958
Validation claims (total / train split / eval split)	1,600 / 800 / 800
Test claims	2,558
Avg. traces per claim (val / test)	15.0 / 20.0
Avg. evidence passages per claim (val / test)	2.84 / 6.11
Validation label dist. (False / Conflicting / True)	57.1% / 23.9% / 19.0%

4. Methodology

4.1. System Overview

Figure 1 shows our three-stage pipeline: (1) **Input construction** - Jaccard-filtered top-5 evidence passages concatenated with the claim and trace; (2) **Scoring** - LoRA-adapted DeBERTa-v3-base with a 3-layer MLP produces a scalar relevance score per trace; (3) **Aggregation** - traces are ranked by score and score-weighted votes determine the final verdict.

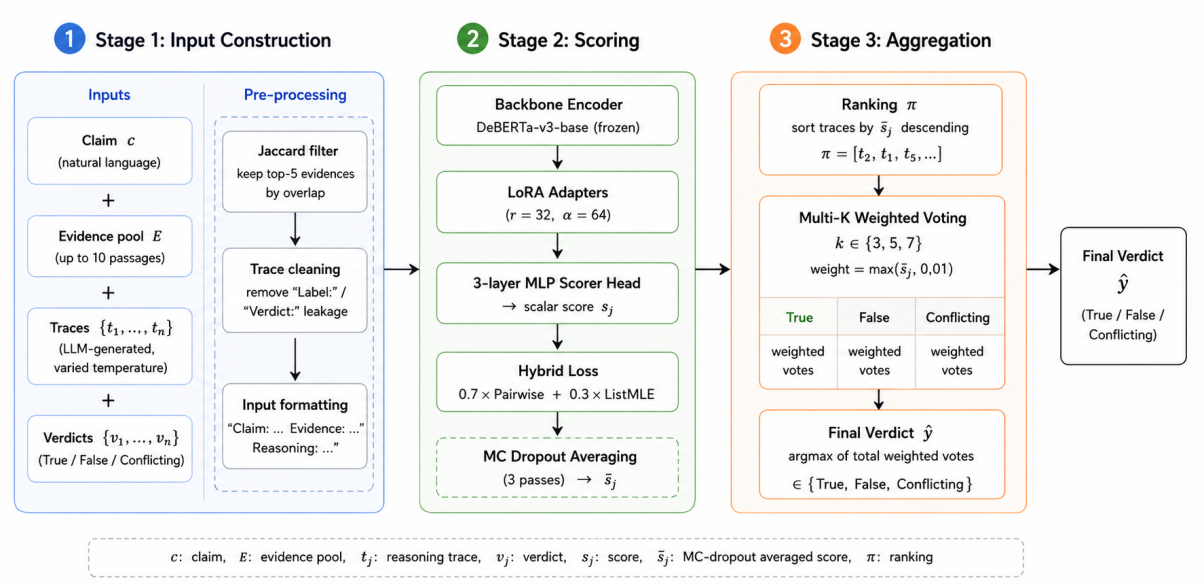

Figure 1: Overview of the PrompterSquad pipeline for CheckThat! 2026 Task 2. **Stage 1** (Input Construction): a claim c, evidence pool E, reasoning traces $\{t_j\}$, and intermediate verdicts $\{v_j\}$ are pre-processed via Jaccard filtering and trace cleaning. **Stage 2** (Scoring): each (claim, evidence, trace) triple is scored by a LoRA-adapted DeBERTa-v3-base with a 3-layer MLP head, using a hybrid pairwise-listwise loss and MC dropout averaging. **Stage 3** (Aggregation): traces are sorted by score to form the ranking output π, and a multi-K weighted voting scheme ($k \in \{3, 5, 7\}$) produces the final verdict $\hat{y}$.

4.2. Input Construction

Each (claim, trace) pair is formatted as:

$$\texttt{Claim: } \{c\} \texttt{ Evidence: } \{e_{i_1}\} \texttt{ [SEP] . . . [SEP] } \{e_{i_5}\} \texttt{ Reasoning: } \{t_j\}$$

Evidence is pre-filtered to the top-5 by Jaccard word-overlap with the claim (parameter-free, no compute overhead), and each passage is truncated to 300 characters to avoid overflowing the 512-token limit. Residual verdict markers (`Label:`, `Verdict:`, SUPPORTS/REFUTES) are removed with a regex pass to prevent label leakage; traces shorter than 5 tokens after cleaning are discarded.

On the choice of Jaccard filtering. We use Jaccard word-overlap deliberately, and it should be read as a *design constraint rather than a claimed contribution.* First, the evidence pool per claim is small (2.84 passages on validation, 6.11 on test; Table 1), so the top-5 filter is inactive on the majority of validation instances and only mildly active on test; the marginal benefit of a heavier retriever is therefore bounded by how often filtering is even triggered. Second, being parameter-free, Jaccard adds no trainable component, which keeps the ablation clean: any measured gain is attributable to the ranker rather than a co-tuned retriever. Third, because the task provides a fixed candidate pool and scores the ranking of *traces* (not passages), evidence selection sits upstream of the metric and influences it only through the 300-character truncation budget. We flag a co-trained or semantic retriever as the most concrete avenue for future gains (Section 7), but chose not to entangle ranker-side improvements with a retrieval change in this submission.

4.3. Model Architecture

We use `microsoft/deberta-v3-base` [10] as a frozen backbone, adapted via LoRA [11] (r=32, α=64, dropout 0.1) on all attention and dense projections ($\sim$1.6M trainable parameters vs. $\sim$184M for full fine-tuning). The `[CLS]` representation is passed through a 3-layer MLP scorer:

$$s(c, E, t) = W_3\,\mathrm{Dropout}_{0.05}(\sigma(W_2\,\mathrm{Dropout}_{0.1}(\sigma(W_1 h_{\mathrm{CLS}} + b_1)) + b_2)) + b_3, \tag{1}$$

where $h_{\mathrm{CLS}} \in \mathbb{R}^{768}$, hidden dims 256 and 128, σ=GELU. Dropout also enables MC averaging at inference.

4.4. Training Objective

We use a hybrid loss combining pairwise margin ranking and ListMLE [17].

Pairwise term. For B (positive, negative) pairs:

$$\mathcal{L}_{\mathrm{PW}} = \frac{1}{B}\sum_{i=1}^{B} \max\bigl(0,\, m - s_i^{+} + s_i^{-}\bigr), \quad m = 0.3. \tag{2}$$

Listwise term. Let π sort items by ground-truth labels (descending). The ListMLE loss is:

$$\mathcal{L}_{\mathrm{LM}} = -\sum_{k=1}^{N-1} \log \frac{\exp(s_{\pi(k)})}{\sum_{j=k}^{N} \exp(s_{\pi(j)})}. \tag{3}$$

This maximises the probability of the ground-truth ordering under the Plackett–Luce model.

Hybrid combination.

$$\mathcal{L} = 0.7\,\mathcal{L}_{\mathrm{PW}} + 0.3\,\mathcal{L}_{\mathrm{LM}}. \tag{4}$$

The pairwise term gives strong per-pair signals; the listwise term injects a global list-aware signal that improves verdict aggregation downstream (confirmed in ablation).

4.5. Self-Distilled Hard Negative Mining

From epoch 2 onwards, at the start of each epoch: (1) score every negative trace with the current model; (2) identify the top-60% highest-scoring negatives per claim as *hard negatives*; (3) mix them equally with randomly sampled negatives for the next epoch's pairs. Starting from epoch 2 (not epoch 1) ensures the model has learned a meaningful ordering before the hard-negative selection is meaningful. No external teacher is needed.

4.6. Optimisation and Training Details

We split the validation set 50/50: half converted to pairwise training pairs (mixed with the JSONL training data), half held out for early stopping. We train with AdamW [13], LoRA learning rate 1×10^{-5} and scorer head 5×10^{-5} (randomly initialised head needs faster adaptation), cosine schedule with 6% warmup, effective batch size 36, gradient clipping at 1.0, FP16. Early stopping triggers after 4 epochs without improvement in $\frac{1}{2}$(Recall@5 + Macro-F1); in practice fires at epoch 7, with best checkpoint at epoch 3.

4.7. Inference

MC dropout averaging. Three forward passes (1 deterministic + 2 with dropout active) are averaged element-wise to produce $\bar{s}_j$, providing a calibrated score estimate [8].

Multi-K weighted voting. The final verdict aggregates score-weighted votes across $k \in \{3, 5, 7\}$:

$$\text{Vote}(\ell) = \sum_{k \in \{3,5,7\}} \sum_{j \in \text{Top}_k} \mathbb{1}[v_j = \ell] \cdot \max(\bar{s}_j, 0.01), \qquad \hat{y} = \arg\max_{\ell} \text{Vote}(\ell). \tag{5}$$

The $\max(\cdot, 0.01)$ clip prevents near-zero scores from being excluded. Note: ablation shows these refinements provide no measurable gain (Section 6.3).

5. Experimental Setup

All experiments are implemented in PyTorch 2.3.1 with Hugging Face `transformers` (4.44.2) and `peft` (0.12.0). Training was carried out on a single **NVIDIA Tesla V100 32 GB** GPU with mixed-precision FP16. A complete run completes in approximately 3.5 hours (early stopping fires at epoch 7). Key hyperparameters: LoRA rank r=32, α=64, dropout 0.1; pairwise margin m=0.3; ListMLE weight λ=0.3; LoRA learning rate 1×10^{-5}, scorer head 5×10^{-5}; effective batch size 36; hard-neg ratio 0.6 starting epoch 2; MC dropout with 3 passes; voting $k \in \{3, 5, 7\}$. All random seeds fixed at 42; the single-seed caveat and its implications for statistical significance are discussed in Section 7.4.

6. Results

6.1. Official Leaderboard

Table 2 reports the top-9 teams on the official English leaderboard. Our system achieves an average score of 0.4138, ranking 9th overall. The gap to the top system is 0.0148 absolute points; the spread within the top-9 is 0.0148, indicating a narrow competitive field. Macro-F1 of our system (0.5967) is competitive with the top scores (range 0.5911–0.6150), while Recall@5 (0.2308) trails the highest score (0.2534) by a wider relative margin.

Why does our system rank last despite a competitive Macro-F1? The low leaderboard position is not a single failure mode but the interaction of a near-saturated task with one specific weakness. On Macro-F1-the harder of the two averaged metrics-we sit mid-pack (rank 6 of 9, only 0.0183 below the top), and our Conflicting-F1 (0.2170) is in fact the third-highest of all teams (Section 6.5). Our deficit is concentrated entirely in Recall@5 (0.2308, the lowest of the 9), which drags down the equally-weighted average. Two factors drive this. (1) Because the average weights Recall@5 and Macro-F1 equally and Macro-F1 differences across the whole field are below 0.024, a $\sim$0.02 Recall@5 gap alone is enough to move a system from mid-pack to last. (2) Our training objective and hard-negative mining optimise verdict-aggregation quality (Macro-F1) more than raw top-5 retrieval: the ablation (Section 6.3) confirms that both training-side components *raise* Macro-F1 while slightly *lowering* Recall@5. We therefore

effectively traded Recall@5 for Macro-F1-favourable for verdict correctness but unfavourable for an equal-weight average in a field this tight. Combined with the structural ceiling of Section 6.5, the 0.0148 separating rank 1 from rank 9 reflects small differences in metric balance rather than a large gap in verifier quality.

Table 2
Official leaderboard for the English track of CheckThat! 2026 Task 2. The PrompterSquad row is our submission.

#	Team	Avg.	Macro-F1	Recall@5	True F1	Conf. F1	False F1
1	ahmadraza123	0.4286	0.6150	0.2421	0.7125	0.2439	0.8888
2	VeriFind	0.4261	0.5988	0.2534	0.7294	0.1788	0.8881
3	GOS-DSGT	0.4234	0.6002	0.2466	0.7400	0.1702	0.8903
4	team-outsider	0.4217	0.5979	0.2456	0.7114	0.1913	0.8909
5	nadie	0.4217	0.6011	0.2424	0.7180	0.1956	0.8896
6	CheckMates	0.4200	0.5948	0.2452	0.7114	0.1866	0.8864
7	mircean2	0.4180	0.6010	0.2350	0.6998	0.2265	0.8767
8	venktesh	0.4175	0.5911	0.2439	0.7154	0.1696	0.8882
9	**PrompterSquad (ours)**	**0.4138**	**0.5967**	**0.2308**	**0.6844**	**0.2170**	**0.8888**

6.2. Per-Class Performance and Baselines

Across all teams: False F1 is uniformly high (0.88–0.89), True is intermediate (0.68–0.74), and Conflicting is the hardest (0.17–0.25) - our Conflicting-F1 (0.2170) ranks third among all teams. On *val_for_eval* (800 claims) our system achieves Recall@5 = 0.4154, Macro-F1 = 0.5132 vs. Recall@5 = 0.2308, Macro-F1 = 0.5967 on test. Against an internal baseline (pairwise loss only, no hard negs, no MC), our final system gains $+0.0058$ ($0.4080 \rightarrow 0.4138$).

On the validation-to-test Recall@5 gap. We attribute the Recall@5 drop ($0.4154 \rightarrow 0.2308$) to the larger test trace pool, and make the mechanism explicit rather than asserting it. Recall@5 measures the fraction of gold-positive traces captured in a *fixed* top-5 window. The window size is constant, but the pool it is drawn from grows from 15.0 traces per claim on validation to 20.0 on test (Table 1). If the number of gold-positive traces per claim is roughly stable across splits, a fixed top-5 covers a smaller *fraction* of positives as the pool grows, so Recall@5 falls even under an unchanged ranker. A first-order estimate illustrates the scale: holding the positive count fixed, moving from a 15- to a 20-trace pool shrinks the expected covered fraction by roughly $15/20 = 0.75\times$, and $0.4154 \times 0.75 \approx 0.31$, moving toward-though not fully reaching-the observed 0.2308. Crucially, this is a property of the metric under a changing denominator, not evidence that the ranker fails to generalise: Macro-F1, which does not depend on a fixed-size window, in fact *increases* from validation (0.5132) to test (0.5967). We therefore read the gap as metric-mechanical rather than a generalisation failure, while acknowledging that a multi-seed, multi-pool study would be required to isolate the effect conclusively (Section 7.4).

6.3. Ablation Study

Table 3 reports three controlled ablations (each trained from scratch for 5 epochs on the same data and seed, evaluated on *val_for_eval*).

ListMLE ($\Delta = -0.0071$ combined when removed) drops Macro-F1 by 0.0249 while Recall@5 slightly improves - the listwise term helps verdict aggregation more than raw top-5 retrieval.

Hard negative mining ($\Delta = -0.0033$) shows the same pattern: Macro-F1 drops, Recall@5 improves when removed. Both training-side components help the verdict task more than the ranking task.

Table 3
Ablation study on the held-out validation half (800 claims). "Final" is our submitted configuration; each subsequent row removes or replaces one component. The combined score is (Recall@5 + Macro-F1)/2. Bold indicates the best value per column. All rows use a single fixed seed (42); see Section 7.4.

Configuration	Recall@5	MRR@5	Macro-F1	Combined
Final (hybrid loss + hard neg + MC + multi-K)	0.4154	0.5839	0.5132	0.4643
− ListMLE (pure pairwise, λ=0)	0.4260	0.5810	0.4883	0.4572
− Hard negative mining (uniform sampling)	0.4247	0.5872	0.4972	0.4610
− MC dropout, − multi-K (deterministic + top-5)	0.4104	**0.5911**	**0.5207**	**0.4656**

MC dropout + multi-K voting (Δ=+0.0013, opposite direction) slightly *hurt* in ablation. These components are therefore not load-bearing; see Section 7.3.

Because all rows share a single seed and the differences between the two strongest configurations ($\pm$0.001–0.007) are of the same order as plausible single-seed noise, we treat the *relative ordering* of the two best rows as indicative rather than statistically established. We therefore draw firm conclusions only for the two larger training-side effects ($\Delta \geq 0.003$: ListMLE and hard-negative mining), and defer significance testing to multi-seed replication (Section 7.4).

6.4. Training Dynamics

Figure 2 shows the per-epoch validation metrics. The combined score peaks at epoch 3 (0.4649) then degrades as training pair-accuracy continues rising (0.69 → 0.77), indicating overfitting. We use the epoch 3 checkpoint for all evaluation.

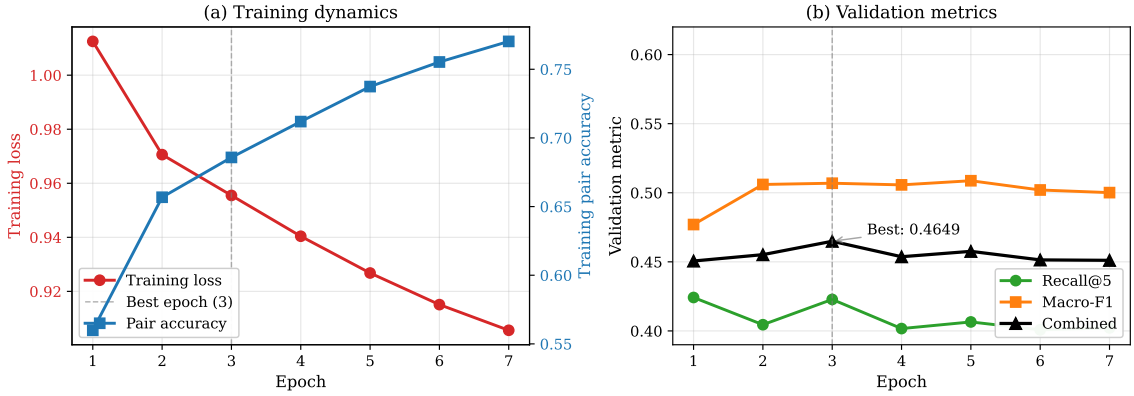

Figure 2: Per-epoch training and validation dynamics of our final configuration on *val_for_eval* (800 claims). (a) Training loss decreases monotonically while training pair accuracy continues to rise from 0.56 to 0.77, indicating that the model is still learning to fit the training distribution. (b) Validation metrics peak at epoch 3 (Combined = 0.4649) and then plateau or degrade slightly, indicating overfitting from epoch 4 onwards. The model checkpoint at epoch 3 is used for all downstream evaluation and the official submission. Early stopping triggers at epoch 7.

6.5. Oracle Upper Bound Analysis

We grant a hypothetical perfect ranker: for each claim with at least one matching trace it places all positives at the top and majority-votes among them; for no-coverage claims it falls back to majority vote over all traces.

Three findings stand out. **The task is nearly saturated**: the oracle Macro-F1 (0.6708) is only 0.056 above the top team, explaining the narrow 0.015-point leaderboard spread. **Conflicting is learnably hard**: the oracle achieves Conflicting-F1 = 0.68, over three times any participating system -

Table 4
Oracle upper-bound on the full validation set (1,600 claims). 471 claims (29.4%) have no matching trace.

Configuration	Macro-F1	True F1	Conf. F1	False F1
Oracle (perfect ranking)	**0.6708**	0.5339	0.6832	0.7954
Top-1 (ahmadraza123)	0.6150	0.7125	0.2439	0.8888
Ours (PrompterSquad)	0.5967	0.6844	0.2170	0.8888

the bottleneck is verifier learning, not data ambiguity. **True is bounded by coverage**: the oracle's True-F1 (0.5339) is paradoxically *lower* than the top team (0.7125) because the fallback majority vote tends to predict False for no-coverage True claims, whereas real systems can exploit learned class priors.

7. Discussion

7.1. Why is the Top-9 Spread So Narrow?

The gap between rank 1 and rank 9 is only 0.0148 on the average score. Our oracle analysis provides a quantitative explanation: the Macro-F1 ceiling for any system using this trace pool is approximately 0.67, and the top system already achieves 0.62. Three structural factors account for this: (1) all teams share the same LLM-generated trace pool, so differences arise only in the verifier; (2) 29.4% of claims have no matching trace for the gold verdict, a floor no verifier can overcome; (3) the dominant False class (57%) yields uniformly high F1, leaving only the harder True/Conflicting classes to differentiate systems.

Improvements in Recall@5 also do not translate linearly to Macro-F1: for no-coverage claims, ranking quality is irrelevant to verdict prediction, and a model that aggressively pushes borderline traces upward can simultaneously raise Recall@5 and lower Macro-F1.

7.2. The Conflicting Class: Learnably Hard

All teams achieve very low Conflicting F1 (0.17–0.25). Our oracle analysis refutes the hypothesis that this class is intrinsically ambiguous: an oracle achieves 0.68 F1 on Conflicting-more than three times the best participating system-showing the bottleneck is verifier learning, not data ambiguity. Two factors explain why real verifiers struggle: (1) pairwise training pairs for Conflicting claims compare Conflicting-positive traces against True/False-labelled negatives, which carry stronger lexical cues, causing the scorer to under-prioritise Conflicting traces; (2) missing even one Conflicting trace from the top-k list flips the majority vote. A dedicated Conflicting-class scorer or entropy-aware aggregator is a clear direction for future work.

7.3. MC Dropout and Multi-K Voting: An Honest Reading

Our ablation shows that removing MC dropout and multi-K voting *improves* validation combined score (0.4643 $\rightarrow$ 0.4656). Three explanations are plausible: (1) the LoRA scorer may already be well-calibrated, making further averaging add noise; (2) top-5 voting aligns directly with the Recall@5 metric, while adding k=3 and k=7 votes may dilute the signal; (3) the differences ($\pm$0.001–0.007) may be within single-seed noise. The honest conclusion is that *these inference-time refinements are not load-bearing contributions.* The dominant gains come from the hybrid loss and hard negative mining. We report this explicitly-including by removing multi-K voting from the camera-ready title-because the field benefits from negative results.

7.4. Limitations

Key limitations include: (1) English-only evaluation; (2) Jaccard evidence filtering is a parameter-free heuristic (Section 4.2)-a co-trained or semantic retriever would likely help and is our most concrete avenue for future gains; (3) no verdict calibration beyond deterministic score-weighted voting; (4) *single-seed evaluation*: all reported numbers use seed 42, with no statistical significance analysis, so small differences-notably the ± 0.001–0.007 gaps in Table 3 and the validation-to-test comparison in Section 6.1-should be read as indicative rather than significant; multi-seed replication with significance testing (e.g. paired bootstrap) is needed to confirm them; (5) ablations are one-at-a-time and do not explore component interactions.

8. Conclusion

We have described the PrompterSquad system for Task 2 of the CLEF 2026 CheckThat! Lab, a fact-checking task on numerical claims under the test-time scaling paradigm. Our approach is a parameter-efficient cross-encoder built on DeBERTa-v3-base with LoRA, trained with a hybrid pairwise-listwise loss, self-distilled hard negative mining, and validation-set mixing. At inference time we apply Monte Carlo dropout averaging and a multi-K weighted voting scheme for verdict aggregation. On the official English test set, our system achieves an average score of 0.4138 and ranks 9th overall in a narrow competitive field; as our oracle analysis shows, this position reflects a metric-balance trade-off (strong Macro-F1, weaker Recall@5) within a near-saturated task rather than a large gap in verifier quality.

A controlled ablation on the held-out validation half identifies the hybrid loss ($\Delta = -0.0071$ when removed) and hard negative mining ($\Delta = -0.0033$ when removed) as the two training-side components contributing meaningful gains, with the listwise term being the larger contributor. The inference-time refinements-Monte Carlo dropout averaging and multi-K weighted voting-do not produce a measurable benefit in the ablation ($\Delta = +0.0013$, in the opposite direction), and we report this honestly rather than presenting them as load-bearing contributions.

Beyond the system itself, our oracle upper-bound analysis uncovers two structural findings that we believe are useful for the community. First, the achievable Macro-F1 ceiling on this trace pool is approximately 0.67, with 29.4% of claims having no trace whose intermediate verdict matches the gold label; this coverage gap explains the unusually narrow 0.015-point spread among the top-9 teams and suggests that future iterations of the task may benefit from a more diverse or larger trace pool. Second, the Conflicting class is *learnably* hard rather than intrinsically ambiguous-an oracle achieves 0.68 F1 on this class versus 0.24 for the best participating system, indicating that targeted methods (e.g. a Conflicting-specialised scorer or entropy-aware aggregator) could in principle close a substantial gap. Future work will explore (i) calibrated verdict voting based on the entropy of the trace verdict distribution, (ii) a dedicated Conflicting-class detector, (iii) a co-trained retriever to replace the current Jaccard filter, and (iv) multi-seed evaluation with significance testing to disambiguate the inference-time refinements.

Code and Data Availability

Our code and trained model checkpoints are publicly released at https://github.com/sonpham811z/PrompterSquad_CheckThat2026 to support reproducibility.

Declaration on Generative AI

During the preparation of this work, the author(s) used Claude (Anthropic) in order to check grammar and spelling and to paraphrase and reword text for clarity, and Perplexity AI in order to assist with literature search and reference discovery. After using these tools/services, the author(s) reviewed and edited the content as needed and take full responsibility for the publication's content. No generative

AI tool is listed as an author, and no AI-generated text, figures, or data were used without human verification.

Acknowledgements

This research was supported by The VNUHCM University of Information Technology's Scientific Research Support Fund.

References

[1] Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024.

[2] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th International Conference on Machine Learning (ICML)*, pages 129–136, 2007.

[3] Jifan Chen, Aniruddh Sriram, Eunsol Choi, and Greg Durrett. Complex claim verification with evidence retrieved in the wild. In *Proceedings of NAACL*, 2024.

[4] Alberto Barrón-Cedeño, Firoj Alam, Tanmoy Chakraborty, Tamer Elsayed, Preslav Nakov, et al. Overview of the CLEF–2024 CheckThat! lab. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction. CLEF 2024 Proceedings*, 2024.

[5] Firoj Alam, Julia Maria Struß, Tanmoy Chakraborty, Preslav Nakov, et al. Overview of the CLEF–2025 CheckThat! lab. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction. CLEF 2025 Proceedings*, 2025.

[6] Julia Maria Struß, Sebastian Schellhammer, Stefan Dietze, Venktesh V., Vinay Setty, Tanmoy Chakraborty, Preslav Nakov, Avishek Anand, Primakov Chungkham, Salim Hafid, Dhruv Sahnan, and Konstantin Todorov. Overview of the CLEF-2026 CheckThat! Lab: Advancing Multilingual Fact-Checking. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction. Proceedings of the Seventeenth International Conference of the CLEF Association (CLEF 2026)*, 2026.

[7] Venktesh V, Vinay Setty, Primakov Chungkham, Avishek Anand, and Firoj Alam. Overview of the CLEF-2026 CheckThat! Lab Task 2 on Fact-Checking Numerical Claims. In *CLEF 2026 Working Notes*, Jena, Germany, 2026.

[8] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of ICML*, 2016.

[9] Luyu Gao, Zhuyun Dai, and Jamie Callan. Rethink training of BERT rerankers in multi-stage retrieval pipeline. In *Proceedings of ECIR*, 2021.

[10] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. DeBERTaV3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*, 2021.

[11] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *Proceedings of ICLR*, 2022.

[12] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *Proceedings of ICLR*, 2024.

[13] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Proceedings of ICLR*, 2019.

[14] Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with BERT. *arXiv preprint arXiv:1901.04085*, 2019.

[15] V Venktesh, Abhijit Anand, Avishek Anand, and Vinay Setty. QuanTemp: A real-world open-domain benchmark for fact-checking numerical claims. In *Proceedings of SIGIR*, 2024.

[16] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *Proceedings of ICLR*, 2023.

[17] Fen Xia, Tie-Yan Liu, Jue Wang, Wensheng Zhang, and Hang Li. Listwise approach to learning to rank: theory and algorithm. In *Proceedings of ICML*, 2008.

[18] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *Proceedings of ICLR*, 2021.

[19] Xinran Zhao, Hongming Zhang, Xiaoman Pan, Wenlin Yao, Dong Yu, Tongshuang Wu, and Jianshu Chen. Verification via probabilistic reasoning over evidence. In *Proceedings of EMNLP*, 2024.