2e1b at CheckThat! 2026: Comparing Pointwise, Pairwise, and Listwise Verifier Objectives Across Model Scales for Numerical Claim Verification

CheckThat! at CLEF 2026

Hasan Berke Bankoglu[1,*,†], Ege Aydin[1,†] and Ege Ozbaran[1,†]

[1]Faculty of Informatics, TU Wien, Favoritenstraße 9-11, 1040 Vienna, Austria

Abstract

We present our submission to CheckThat! 2026 Task 2, which focuses on numerical claim verification using reasoning-trace ranking. Given a claim, evidence, and a set of Large Language Model (LLM)-generated reasoning traces with candidate verdicts, the system ranks traces by utility and derives a final verdict from the top-ranked ones. We investigate whether the best verifier training objective remains consistent across model scales by fine-tuning Qwen3-Base models at three sizes (0.6B, 1.7B, and 4B) with Low-Rank Adaptation (LoRA) adapters and a scalar verifier head, varying only the training objective: pointwise Binary Cross-Entropy (BCE), pairwise RankNet, or listwise LambdaLoss. On English validation, pointwise BCE offers a slight advantage at the two smaller scales that disappears at 4B, while listwise training consistently underperforms. Our best validation result is achieved by an ensemble of two 4B verifiers trained with different objectives, outperforming all single-objective configurations. English-trained verifiers are also evaluated zero-shot on Spanish and Arabic, where the submitted systems exceed the released organizer baselines, while the English test result falls below the reference baseline despite stronger validation performance. These findings indicate that the best training objective depends on model scale rather than holding across all sizes, and that combining verifiers trained with different objectives is more effective than combining verifiers that differ only in scale.

Keywords

Numerical claim verification, verifier models, ranking objectives, cross-lingual transfer, LoRA fine-tuning

1. Introduction

Numerical claim verification involves evaluating factual statements that rely on quantities, dates, intervals, and temporal relations, which frequently cannot be validated through surface-level entity matching against evidence [1]. CheckThat! 2026 Task 2 conceptualizes this as reasoning-trace verification: for each claim, systems receive evidence and a fixed set of Large Language Model (LLM)-generated reasoning traces, each associated with a candidate verdict [2, 3]. Systems are then required to rank these traces based on their utility for deriving the correct verdict and subsequently predict a final claim label from the highest-ranked traces. This setting extends the CheckThat! 2025 [4] numerical claim verification task by inserting reasoning-trace ranking as an intermediate step between evidence and the final verdict.

The task therefore decomposes into two coupled subproblems: scoring each candidate's reasoning trace and aggregating the top-ranked traces to produce a verdict. This study concentrates on the first subproblem, namely the verifier-training component of the pipeline. The central research question is whether the optimal learning-to-rank objective remains consistent across different model scales. To investigate this, Qwen3-Base models of three sizes (0.6B, 1.7B, and 4B parameters) are fine-tuned using a unified Low-Rank Adaptation (LoRA) verifier architecture and inference procedure, varying only the

CLEF 2026 Working Notes, 21 – 24 September 2026, Jena, Germany

*Corresponding author.

†These authors contributed equally.

e12432802@student.tuwien.ac.at (H. B. Bankoglu); ege.aydin@tuwien.ac.at (E. Aydin); e12433722@student.tuwien.ac.at (E. Ozbaran)

0009-0003-5267-1594 (H. B. Bankoglu); 0009-0006-3019-2412 (E. Aydin); 0009-0001-8156-155X (E. Ozbaran)

training objective: pointwise Binary Cross-Entropy (BCE), pairwise RankNet, or listwise LambdaLoss.

This paper makes three primary contributions: (i) a controlled 3×3 comparison of learning-to-rank objectives across model scales for reasoning-trace verifier training; (ii) an exploration of objective-diverse ensembling as an alternative to scale-diverse ensembling; and (iii) a zero-shot cross-lingual evaluation of English-trained verifiers on Spanish and Arabic, without in-domain fine-tuning, translation, or external resources. The remainder of the paper is organized as follows: Section 2 reviews background and related work, Section 3 describes the task and dataset, Section 4 presents the methodology, Section 5 specifies the experimental setup, Section 6 reports results and analysis, and Section 7 concludes.

2. Background and Related Work

Numerical claim verification. Numerical claim verification is a specialized form of automated fact-checking in which the truth of a claim depends on quantities, dates, comparisons, intervals, or temporal relations. QuanTemp introduced a real-world benchmark for this task and demonstrated that numerical claim verification remains challenging even for advanced Natural Language Inference (NLI) and LLM-based systems [1]. The CheckThat! lab has established numerical claim verification as a recurring shared-task setting. The 2025 edition framed it as a three-way classification of claims into true, false, or conflicting, given retrieved evidence [5]. The 2026 edition (Task 2) [2] extended this setup by providing LLM-generated reasoning traces as intermediate candidates that must be ranked before deriving the final verdict. Systems submitted in the 2025 edition primarily focused on evidence retrieval, context construction, tokenization strategies, and direct veracity classification. For example, DS@GT studied ModernBERT-based numerical fact verification with longer input contexts and right-to-left numerical tokenization, finding that neither longer context nor modified tokenization resolved the underlying evidence-quality bottleneck [6].

Verifier models for LLM reasoning. Another line of research trains scalar models to score LLM-generated reasoning paths and select among them. Cobbe et al. [7] introduced verifier models for solving math word problems, training a classifier to rank candidate solutions sampled from a generator. Lightman et al. [8] distinguished between outcome reward models, which score complete reasoning paths, and process reward models, which score intermediate steps. They demonstrated the advantage of process reward models for multi-step mathematical reasoning. In the fact-checking context, Chungkham et al. [9] applied verifier-guided test-time scaling to numerical claim verification by sampling multiple LLM reasoning paths and training a custom verifier to select the path most likely to lead to the correct verdict. Reasoning-trace evaluation has also been studied as a measurement problem. Lee and Hockenmaier [10] organize evaluation criteria into factuality, validity, coherence, and utility, with utility corresponding to whether a complete trace supports reaching the correct answer.

Learning-to-rank objectives. The comparison of pointwise, pairwise, and listwise ranking objectives has a long history in document ranking. RankNet [11] is a canonical pairwise objective, framing ranking as binary classification of preference pairs. LambdaLoss [12] extends this approach to listwise training by optimizing weighted approximations of ranking metrics such as Normalized Discounted Cumulative Gain (NDCG). TFR-BERT [13] provided one of the earliest comparisons of pointwise, pairwise, and listwise objectives using pretrained transformer representations on standard document-ranking benchmarks. More recent work has revisited this comparison with larger encoder-decoder backbones. For example, RankT5 [14] compares pointwise and listwise ranking losses on T5-scale models and reports objective-dependent gains that vary with model size, which aligns with the question addressed in this paper.

3. Task and Dataset

CheckThat! 2026 Task 2 [2] focuses on the fact-checking of numerical claims using reasoning-trace verification. Each instance comprises a naturally occurring claim involving quantities or temporal expressions, associated evidence, and a fixed set of reasoning traces generated by LLM, each with a candidate verdict. Participating systems are required to score the candidate traces, rank them based on their utility for determining the correct verdict, and derive a final claim-level verdict from the highest-ranked traces. Figure 1 illustrates this pipeline: the verifier assigns a scalar score to each of the 20 reasoning traces, and the final verdict is obtained by majority vote over the candidate verdicts of the top-5 highest-scoring traces.

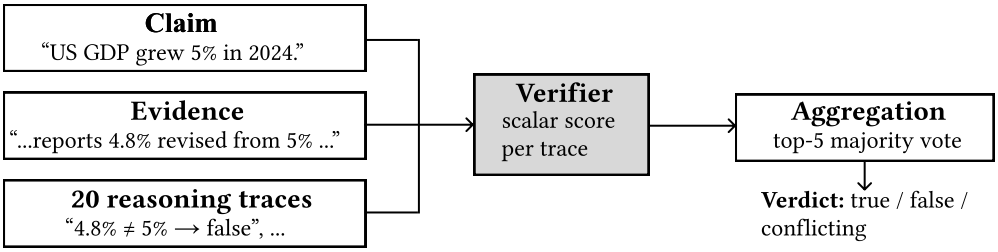

Figure 1: CheckThat! 2026 Task 2 pipeline.

The task encompasses three languages: English, Spanish, and Arabic. The English dataset is derived from QuanTemp [1], while the Spanish and Arabic datasets originate from the previous CheckThat! numerical claim verification edition [5, 4], each with corresponding evidence collections. For every claim, the organizers provide 20 generated reasoning traces, the evidence corpus, and top-ranked evidence snippets, enabling participants to construct retrieval or reranking pipelines if desired. The English and Spanish splits employ a ternary verdict label set (*true*, *false*, *conflicting*), whereas the Arabic split utilizes a binary label set (*true*, *false*).

Table 1 presents the per-language train and validation splits used in this study. The test splits are withheld by the organizers and evaluated using Codabench.

Table 1
Dataset statistics for the official train/validation splits of CheckThat! 2026 Task 2 [2]. Each claim is paired with 20 reasoning traces.

Language	Train claims	Validation claims	Total claims	Verdict labels
English	6,400	1,600	8,000	true / false / conflicting
Spanish	2,246	562	2,808	true / false / conflicting
Arabic	2,608	652	3,260	true / false

In the pointwise formulation adopted in this study, each claim–trace pair constitutes a single training row, resulting in 128,000 English training rows and 32,000 validation rows. The corresponding Spanish and Arabic counts are provided in Table 1. Pairwise and listwise formulations, which group rows by claim, are detailed in Section 4.

The official evaluation protocol integrates both ranking and classification performance. Trace ranking is assessed using Recall@5 over the ranked trace list, while claim verification is evaluated using macro-averaged F_1 over the predicted final verdict. The official combined score, which is used by the organizers for system ranking and throughout this study, is the unweighted mean of these two metrics.

$$\text{combined} = \frac{\text{Recall@5} + \text{macro-}F_1}{2} \tag{1}$$

4. Architecture

Our verifier-based ranking pipeline assigns one scalar score to each candidate reasoning trace and derives the final claim verdict by aggregating the top-ranked traces. To isolate the effect of training objective from other design choices, the verifier architecture, input format, preprocessing, and inference procedure are held constant across the main experimental matrix; only the Qwen3-Base backbone scale and the training objective are varied. The verifier itself is described in Section 4.1, the three training objectives in Section 4.2, and the inference procedure in Section 4.3.

4.1. Verifier

Each verifier uses a Qwen3-Base backbone [15] with LoRA adapters [16], followed by a scalar linear head. Three backbone sizes are evaluated: 0.6B, 1.7B, and 4B parameters. Base models are selected instead of instruction-tuned or reranker-specific variants to prevent confounding the comparison with instruction tuning or retrieval-oriented pretraining. Figure 2 presents the verifier internals.

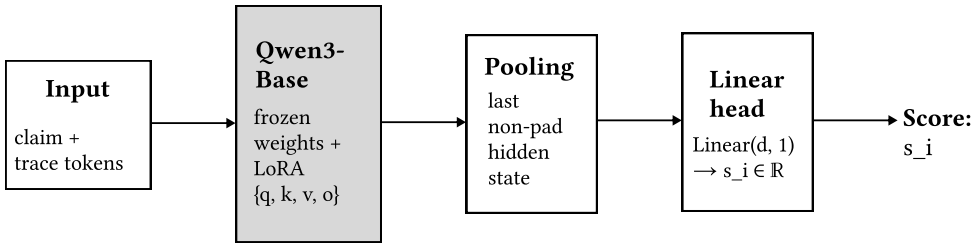

Figure 2: Verifier architecture.

The input consists of the concatenation of the claim and one candidate reasoning trace, which is tokenized and right-padded to a maximum sequence length of 1024 tokens. An `evidence_off` input mode is used, in which the verifier does not append the raw evidence text. Preliminary experiments indicated that concatenating raw evidence degraded validation performance, as the generated reasoning traces already include evidence-derived numerical and factual cues.

The tokenized sequence is processed by the Qwen3-Base backbone, with its weights kept frozen. LoRA adapters are applied to the attention projection matrices $\{q, k, v, o\}$ with rank $r = 16$, scaling factor $\alpha = 32$, and dropout 0.05. The final hidden state at the last non-padding token is pooled and input to a scalar linear head:

$$s_i = w^\top h_i + b \tag{2}$$

where h_i is the pooled representation of trace i and s_i is its verifier score.

4.2. Training Objectives

A reasoning trace is considered useful if its candidate verdict matches the gold claim verdict. A binary trace label $y_i \in \{0, 1\}$ is defined, where $y_i = 1$ indicates that trace i leads to the correct verdict. Three learning-to-rank objectives are compared.

Pointwise binary cross-entropy. The pointwise model treats each claim–trace pair independently and optimizes the binary cross-entropy loss:

$$\mathcal{L}_{\text{point}} = -\sum_i \left[y_i \log \sigma(s_i) + (1 - y_i) \log(1 - \sigma(s_i)) \right] \tag{3}$$

Each trace contributes a single labeled training example, resulting in dense supervision.

Pairwise RankNet. The pairwise model constructs positive–negative trace pairs within the same claim. For each sampled pair (i^+, i^-) with scores s^+, s^-, the RankNet loss [11] is defined as

$$\ell_{\text{pair}}(i^+, i^-) = -\log \sigma(s^+ - s^-) \tag{4}$$

The per-claim loss is computed as the sum over its sampled pairs. The number of positive–negative pairs per claim is limited to $M = 8$, sampled in a deterministic order using a fixed seed.

Listwise LambdaLoss. The listwise model is trained on trace groups for each claim and optimizes LambdaLoss [12], specifically the NDCG-aware variant. Each listwise example consists of a trace group of size $G = 20$, corresponding to the per-claim trace count in the released data.

4.3. Inference and Verdict Aggregation

During inference, all three objective variants employ the same procedure. The verifier scores each candidate trace independently, and the top five traces by score are selected. The final claim verdict is determined by majority vote over their candidate verdicts. Trace verdicts outside the allowed label set for the corresponding language are filtered prior to voting. In the event of a tie, the verdict of the highest-scoring trace among the tied labels is used.

5. Experimental Setup

The experiments in this work isolate the effect of two design choices in verifier training: backbone scale and training objective. All other hyperparameters, including optimizer, learning rate, schedule, LoRA configuration, batching, and inference procedure, remain constant across experiments. This section details the shared training protocol, evaluation methodology, baselines and ablations contextualizing the matrix results, and the systems submitted to the official leaderboard.

5.1. Training Configuration

Table 5 in Appendix A summarizes the optimization, batching, and architectural hyperparameters shared across all experiments, including a per-cell training duration of 5 epochs.

5.2. Evaluation Metrics

The official Task 2 [2] evaluation protocol, described in Section 3, is followed. Mean Reciprocal Rank (MRR)@k is designated in the official instructions as an internal ranking metric but is excluded from the combined score and is not used as an optimization target. Validation results in Section 6 are computed using the official scorer.

5.3. Baselines and Ablations

Organizer baseline. The organizers provide a reference verifier based on LLaMA-3.2-1B, fine-tuned with LoRA on the released training data. On the English validation split, this verifier achieves a combined score of 0.4246, serving as a reference point for validation results in Section 6.

Internal baseline (B0). Prior to the Qwen3 matrix experiments, a ModernBERT-large verifier was trained using pointwise binary cross-entropy as an internal baseline. This baseline assesses whether an encoder-only verifier can generate effective trace rankings and establishes a lower bound for subsequent comparisons; it reaches a combined score of 0.3528 on the English validation split.

Objective × scale matrix. The primary comparison utilizes a 3×3 matrix that combines three backbone scales (Qwen3-Base 0.6B, 1.7B, and 4B) with three training objectives: pointwise binary cross-entropy, pairwise RankNet, and listwise LambdaLoss.

Ablations. Four ablation studies examine specific design choices and characterize the robustness of the matrix results:

- **Evidence-on**: the verifier is given the raw evidence text concatenated to the claim and trace, testing whether direct evidence access helps or instead adds redundant noise.
- **Off-task starters**: two non-Base 4B starting points are fine-tuned under the same pointwise recipe, testing whether retrieval-oriented pretraining (Qwen3-Reranker-4B) and a different model family (Gemma-3-4B) transfer to reasoning-trace verification.
- **Ensemble axis**: an objective-diverse ensemble (4B pointwise + 4B pairwise) is compared against a scale-diverse ensemble (1.7B pointwise + 4B pointwise) to isolate the source of the ensemble gain.
- **Reproducibility rerun**: the matrix is rerun after a routine library upgrade (transformers, torch, and peft versions updated between the original and rerun passes). This rerun characterizes version sensitivity; the primary results in Section 6 are from the original training pass, and the rerun also contributes the Qwen3-4B pairwise verifier to the submitted English ensemble.

5.4. Submitted Systems

For English, the submitted ensemble averages the score lists from two 4B verifiers: the Qwen3-4B pointwise model from the original training pass and the Qwen3-4B pairwise RankNet model from the rerun pass. The averaged scores are aggregated using a top-5 majority-vote approach. For Spanish and Arabic, the submitted systems employ zero-shot cross-lingual transfer from English-trained checkpoints, averaging the score lists of the Qwen3-1.7B and Qwen3-4B pointwise verifiers, both trained on English. No fine-tuning, translation, or external data is used for Spanish or Arabic.

6. Results and Discussion

We present our results by answering four main questions: (i) what the 3×3 objective × scale comparison shows on English validation; (ii) how the submitted systems perform on the official held-out test sets; (iii) which ablations matter; and (iv) where the main errors concentrate.

6.1. Validation Results

Table 2 reports the comparison on the English validation split under the combined score of Eq. 1. Figure 3 in Appendix B visualizes the same objective-by-scale comparison. All configurations use the `evidence_off` input mode and the training recipe of Section 5.

Table 2
Qwen3-Base objective × scale comparison on English validation. Best per row in bold, best overall underlined.

Backbone	Pointwise BCE	Pairwise RankNet	Listwise LambdaLoss
Qwen3-0.6B	**0.4205**	0.4092	0.4054
Qwen3-1.7B	**<u>0.4222</u>**	0.4180	0.4114
Qwen3-4B	0.4208	**0.4209**	0.4163

Pointwise performs best at smaller scales, while pairwise matches its performance at 4B. Pointwise binary cross-entropy gives the highest combined score at 0.6B and 1.7B, outperforming pairwise RankNet by 0.0113 and 0.0042. At 4B, the two objectives are almost equal, with only a 0.0001 difference. Since all models use a single seed, we do not consider the 0.0042 gap at 1.7B significant due to retraining noise. Overall, pointwise is at least as strong as pairwise at every scale, with the biggest difference at 0.6B and no difference at 4B. This does not suggest a consistent trend based on scale.

Listwise performs worse than the other objectives at every scale. Listwise LambdaLoss is the weakest objective at all three sizes, trailing the strongest setup by about 0.012 to 0.015 combined-score points. These differences are much larger than those between pointwise and pairwise, and are probably not just due to seed noise. Listwise scores do improve as the model size increases (0.4054, 0.4114, 0.4163), which suggests that listwise training might work better with larger models or a different optimization budget. However, it is still the weakest in our current setup.

Under our fixed budget, the best single configuration is 1.7B. Qwen3-1.7B with pointwise BCE achieves the highest single combined score (0.4222), just 0.0014 higher than the best 4B setup. This small difference could be due to seed variation, so we do not claim that 1.7B is truly the best scale. Since all setups use the same number of epochs, learning rate, and batch size, it is possible that the fixed budget works better for smaller models than for 4B. Adjusting the training for scale could change these results.

Ensembling improves validation performance. A score-list mean of the two 4B verifiers (pointwise BCE and pairwise RankNet) gives 0.4375 on English validation. This is 0.0129 higher than the organizer's validation baseline of 0.4246 and at least 0.0153 better than any single configuration. This ensemble combines two objectives at the same scale. The ablations in Section 6.3 compare this objective-diverse ensemble to a scale-diverse one to see if the improvement comes from using different objectives.

6.2. Official Test Results

We submitted one run per language to the official Codabench leaderboard, selecting each submission by validation combined score. Table 3 reports the leaderboard-visible scores for our selected submissions and the released organizer baselines. The English submission is the objective-diverse 4B ensemble; the Spanish and Arabic submissions use zero-shot transfer from English-trained Qwen3 checkpoints.

Table 3
Official Codabench test results. Spanish and Arabic use English-trained checkpoints in a zero-shot cross-lingual setting. Δ is the difference from the organizer baseline.

Language	Our submission	Organizer baseline	Δ
English	0.4082	**0.4175**	-0.0093
Spanish	**0.4094**	0.2874	$+0.1220$
Arabic	**0.5788**	0.2963	$+0.2825$

English: validation-based selection did not transfer. On English, the selected ensemble falls 0.0093 below the organizer baseline on test, despite exceeding the organizer's validation baseline by 0.0129. The submission was selected by validation combined score, and a different submitted run — not selected because it scored lower on validation — reached 0.4144 on test, above our selected run. The validation-to-test reordering indicates that validation combined score was not a reliable selection criterion for held-out test performance under our setup.

Spanish and Arabic: strong zero-shot transfer. Both cross-lingual submissions substantially exceed the organizer baselines (Table 3) using only English-trained checkpoints, with no in-language fine-tuning, translation, or external resources. Two factors temper the size of these gains. First, the organizer baseline is an English-trained reference verifier and is not optimized for cross-lingual transfer, so it is a weak comparison point on Spanish and Arabic. Second, the Arabic label set is binary (Section 3), so macro-F_1 on Arabic is computed over two classes rather than three, which raises absolute combined scores relative to the ternary English and Spanish settings. The Arabic improvement remains the largest among our submissions after accounting for these factors.

6.3. Ablation Results

Evidence-on hurts. Concatenating the raw evidence text to the claim and trace (evidence-on) reduced the combined score in the pointwise ablations we measured: from 0.4156 to 0.3962 at 0.6B (-0.0194) and from 0.4126 to 0.3952 at 1.7B (-0.0174). The released traces are generated with evidence already in context, so feeding the same evidence a second time appears to add redundant or distracting tokens within the 1024-token budget rather than introducing a new signal.

Objective diversity beats scale diversity in ensembling. Table 4 compares score-mean ensembles on English validation. An objective-diverse ensemble at fixed 4B scale (4B pointwise + 4B pairwise) reaches 0.4375, while a scale-diverse ensemble at fixed pointwise objective (1.7B pointwise + 4B pointwise) reaches 0.4251. The 0.0124 gap suggests that pointwise and pairwise verifiers make more complementary errors than verifiers of different scales trained under the same objective.

Table 4
Score-mean ensembles on English validation. P denotes pointwise, PW pairwise.

Ensemble	Diversity axis	Combined
4B-P + 4B-PW (submitted, rerun)	objective	**0.4375**
4B-P + 4B-PW (original)	objective	0.4365
1.7B-P + 4B-P	scale	0.4251
0.6B-P + 1.7B-P + 4B-P + 4B-PW	both	0.4360

Off-task starters. We also tested non-Base starting points under the same pointwise recipe. Qwen3-Reranker-4B did not improve over Qwen3-4B-Base, suggesting that retrieval-oriented pretraining does not transfer cleanly to reasoning-trace verification. Gemma-3-4B was trained and evaluated as an alternative 4B-scale backbone but did not enter the submitted ensemble. We report these as ablations rather than main systems because the controlled objective–scale comparison is based on Qwen3-Base models.

Rerun stability. Retraining under the upgraded library stack (Section 5.3) produced a systematic regression of roughly 0.006 combined score relative to the original pass. We therefore use the original-pass numbers as the primary validation results and report rerun numbers only where they materially affect the submitted ensemble.

6.4. Error Analysis

We analyze the three 4B-scale single systems (pointwise, pairwise, and listwise) and the submitted ensemble on the 1,600-claim English validation split.

Where the ensemble helps. The ensemble does not improve Recall@5 substantially over the best single system (0.3224 vs. 0.3198, $+0.0026$); most of its gain comes from macro-F_1 (0.5535 vs. 0.5228,

$+0.0307$). The ensemble does not find more correct traces in the top five so much as it derives a better verdict from the ranked traces.

Per-class breakdown. The minority *true* class is the hardest for every system (F_1 around 0.42–0.44 for single systems, 0.46 for the ensemble), and the *conflicting* class is similarly weak (0.43–0.45 for single systems, 0.48 for the ensemble). The dominant *false* class is the strongest (≈ 0.68 for single systems, 0.72 for the ensemble). The ensemble's macro-F_1 gain is therefore concentrated in the two minority classes.

Score separation. We define score separation as the mean verifier score on correct traces minus the mean on incorrect traces, in the raw (pre-sigmoid) score units of each system. The pointwise system produces the largest separation (10.78), the pairwise system is intermediate (7.62), and the listwise system is the most compressed (1.30). This is consistent with the training signals: pointwise BCE pushes absolute scores apart, RankNet constrains only pairwise differences, and LambdaLoss enforces mainly relative order within a group.

Recall@5 versus macro-F_1. Across all configurations, Recall@5 varies less than macro-F_1. Differences between objectives appear less in whether a correct trace reaches the top five than in how the top-ranked verdicts are distributed and aggregated. In practice, scaling and ensembling help the verdict-aggregation step more than the top-5 retrieval step.

7. Conclusion and Future Work

We examined whether the optimal learning-to-rank objective for reasoning-trace verification remains consistent across backbone scales in CheckThat! 2026 Task 2 [2]. With the verifier architecture, input format, optimization setup, and inference procedure held constant, we compared three Qwen3-Base scales (0.6B, 1.7B, 4B) using three objectives: pointwise binary cross-entropy, pairwise RankNet, and listwise LambdaLoss. On English validation, pointwise achieves the highest combined score at the two smaller scales, but its advantage over pairwise narrows to nothing at 4B, and listwise trails at every scale. The strongest validation result is not a single model but an objective-diverse ensemble combining 4B pointwise and 4B pairwise verifiers, which outperforms a scale-diverse ensemble. This suggests that objective diversity provides more effective error decorrelation than scale diversity at a fixed budget.

These findings have two main limitations. First, each configuration was trained with a single seed, so we do not claim significance for differences smaller than 0.005 among the top results. Second, all scales used the same fixed training budget, which may favor smaller models and partly explain why the best single configuration is 1.7B rather than 4B. On the official test sets, these limitations are evident: the English ensemble does not maintain its validation advantage and falls slightly below the organizer baseline, while the Spanish and Arabic zero-shot submissions exceed their baselines using only English-trained checkpoints. Validation improvements are meaningful but do not consistently transfer across held-out splits and languages.

Several future directions emerge. Scale-aware hyperparameter tuning could determine whether the observed non-monotonic scale behavior persists with per-scale budgets. Listwise training may improve with alternative losses, enhanced group sampling, or larger trace pools than the fixed 20 used here. The current ensemble averages raw score lists; learned calibration or meta-aggregation could better leverage complementary objectives. Finally, the gap between validation and test results highlights the need for more detailed robustness analysis, including per-claim-type evaluation and clear guidelines for when to provide raw evidence directly to the verifier.

Declaration on Generative AI

During the preparation of this work, the author(s) used Claude Opus 4.7 and Grammarly for grammar and clarity revision. These tools were employed to refine sentence structure, correct typographical errors, and improve overall language quality. No generative content was used for analysis, figures, or experimental sections. After using these tool(s)/service(s), the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] V. V, A. Anand, A. Anand, V. Setty, Quantemp: A real-world open-domain benchmark for fact-checking numerical claims, in: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '24, Association for Computing Machinery, New York, NY, USA, 2024, p. 650–660. doi:10.1145/3626772.3657874.

[2] V. V, V. Setty, P. Chungkham, A. Anand, F. Alam, Overview of the CLEF-2026 CheckThat! lab task 2 on fact-checking numerical claims, CLEF 2026, Jena, Germany, 2026.

[3] J. M. Struß, S. Schellhammer, S. Dietze, V. V, V. Setty, T. Chakraborty, P. Nakov, A. Anand, P. Chungkham, S. Hafid, D. Sahnan, K. Todorov, The CLEF-2026 CheckThat! lab: Advancing multilingual fact-checking, 2026. arXiv:2602.09516.

[4] V. Venktesh, V. Setty, A. Anand, M. Hasanain, B. Bendou, H. Bouamor, F. Alam, G. Iturra-Bocaz, P. Galuščáková, Overview of the CLEF-2025 CheckThat! lab task 3 on fact-checking numerical claims, in: G. Faggioli, N. Ferro, P. Rosso, D. Spina (Eds.), Working Notes of CLEF 2025 - Conference and Labs of the Evaluation Forum, CLEF 2025, Madrid, Spain, 2025.

[5] 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: Experimental IR Meets Multilinguality, Multimodality, and Interaction, volume 16089 of *Lecture Notes in Computer Science*, Springer Nature Switzerland, 2026, pp. 199–223. doi:10.1007/978-3-032-04354-2_13.

[6] M. Heil, A. Pramov, Ds@gt at checkthat! 2025: Evaluating context and tokenization strategies for numerical fact verification, in: Working Notes of CLEF 2025 - Conference and Labs of the Evaluation Forum, 2025.

[7] K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, et al., Training verifiers to solve math word problems, 2021. arXiv:2110.14168.

[8] H. Lightman, V. Kosaraju, Y. Burda, H. Edwards, B. Baker, T. Lee, J. Leike, J. Schulman, I. Sutskever, K. Cobbe, Let's verify step by step, in: The Twelfth International Conference on Learning Representations, 2024.

[9] P. Chungkham, V. V, V. Setty, A. Anand, Think right, not more: Test-time scaling for numerical claim verification, in: Findings of the Association for Computational Linguistics: EMNLP 2025, 2025, pp. 24345–24363. doi:10.18653/v1/2025.findings-emnlp.1322.

[10] J. Lee, J. Hockenmaier, Evaluating step-by-step reasoning traces: A survey, in: Findings of the Association for Computational Linguistics: EMNLP 2025, 2025, pp. 1789–1814. doi:10.18653/v1/2025.findings-emnlp.94.

[11] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, G. Hullender, Learning to rank using gradient descent, in: Proceedings of the 22nd International Conference on Machine Learning (ICML), 2005, pp. 89–96.

[12] X. Wang, C. Li, N. Golbandi, M. Bendersky, M. Najork, The LambdaLoss framework for ranking metric optimization, in: Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM), 2018, pp. 1313–1322.

[13] S. Han, X. Wang, M. Bendersky, M. Najork, Learning-to-rank with BERT in TF-Ranking, 2020. arXiv:2004.08476.

[14] H. Zhuang, Z. Qin, R. Jagerman, K. Hui, J. Ma, J. Lu, J. Ni, X. Wang, M. Bendersky, Rankt5:

Fine-tuning t5 for text ranking with ranking losses, in: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2023.
[15] Qwen Team, Qwen3 technical report, 2025. arXiv:2505.09388.
[16] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, LoRA: Low-rank adaptation of large language models, in: International Conference on Learning Representations (ICLR), 2022.
[17] I. Loshchilov, F. Hutter, Decoupled weight decay regularization, in: International Conference on Learning Representations (ICLR), 2019.

A. Hyperparameters

Table 5 lists the complete set of optimization, batching, and architecture hyperparameters used across all experiments in this work.

Table 5
Training and architecture hyperparameters. Values are shared across experiments unless noted.

Optimization	
Optimizer	AdamW [17]
Learning rate	1×10^{-4}
Weight decay	0.01
Schedule	Linear warmup, ratio 0.06
Precision	bf16 backbone, fp32 head
Epochs	5; 0.6B pairwise/listwise: 3
Batching and sequence	
Per-device batch size	8
Gradient accumulation	8
Effective batch size	64
Evaluation batch size	16
Max sequence length	1024 tokens; right-padded/truncated
LoRA	
Target modules	Attention projections q, k, v, o
Rank r	16
Scaling factor α	32
Dropout	0.05
Objective-specific	
Pairwise pair cap M	8 per claim
Listwise group size G	20
Hardware and reproducibility	
GPU	1 NVIDIA A40 48 GB per run
Seed	42

B. Results Visualization

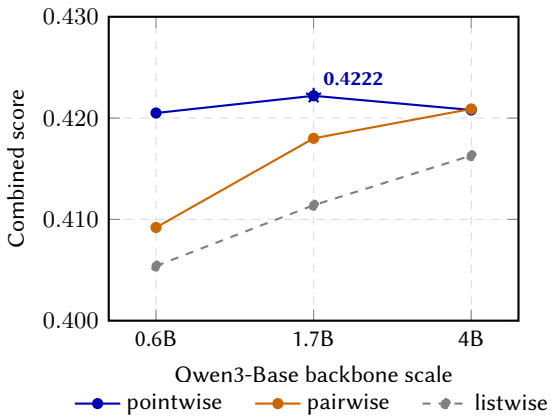

Figure 3: Combined-score comparison across backbone scales and training objectives on the English validation split. The star marks the best single configuration (Qwen3-1.7B with pointwise BCE).