# Seals-NLP at Touché: Comparing Fully Tuned Encoders and Optimized LLMs for Multi-Task Fallacy Detection

Touché at CLEF 2026

Minh Smith[1,*], Minarul Islam[1] and Cheryl Seals[1]

[1]Department of Computer Science and Software Engineering, Auburn University, Auburn, AL, USA

**Abstract**

This paper presents the Seals-NLP submission to the Touché 2026 Fallacy Detection shared task. We systematically compare fully fine-tuned transformer encoders against parameter-efficient large language models (LLMs) on fallacy presence, type, and argument-scheme classification, over both raw and rhetorically enhanced argument text. Fully fine-tuned encoders remain the strongest systems on raw text, with ModernBERT-large leading binary detection and type classification. The most decisive factor, however, is the input format: explicitly articulating latent reasoning dramatically improves every architecture. Our submitted run, a RoBERTa-large fine-tuned on the enhanced text, ranked first on the enhanced track of Subtask 1 in the official leaderboard, and fourth on Subtasks 2 and 3.

**Keywords**

Fallacy Detection, Argument Mining, Multi-Task Learning, Large Language Models, Contrastive Learning, Parameter-Efficient Fine-Tuning, Touché 2026

## 1. Introduction

A logical fallacy is an argument whose premises fail to adequately support its conclusion. The Touché 2026 Fallacy Detection task challenges systems to identify and categorize this flawed reasoning across three subtasks [1, 2]: Subtask 1 (Fallacy Detection) determines whether an argument is fallacious; Subtask 2 (Fallacy Classification) identifies the specific fallacy type; and Subtask 3 (Scheme Classification) identifies the underlying argument scheme.

The dataset provides two textual views for each argument: a base form and a rhetorically enhanced form that explicitly articulates implicit reasoning. We leverage this structure to compare traditional transformer encoders trained via full fine-tuning against parameter-efficient large language models (LLMs) adapted via prompting or supervised-contrastive learning [3, 4], quantifying the exact impact of explicit rhetorical enrichment across both text fields.

Our primary contributions are: (i) a systematic comparison of fully fine-tuned encoders against adapted LLMs on all subtasks; and (ii) an official submission (RoBERTa-large on enhanced text) that ranked first on the enhanced track of Subtask 1. Our code and artifacts are publicly available.[1]

## 2. Background

**Fallacy detection and argument mining.** Computational fallacy detection has evolved from foundational argument mining to identifying manipulative rhetoric and logical flaws, reflected in recent benchmarks such as Jin et al.'s dataset [5] and the FAINA corpus [6]; the CLEF Touché series has driven the domain through argumentation shared tasks since 2020 [7]. Two model families dominate the resulting classification problem: compact bidirectional encoders, which excel at short-text classification, and autoregressive LLMs, which offer zero-shot capacity and, when pretrained for it, high-quality text

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

*Corresponding author.

mzs0193@auburn.edu (M. Smith); mzi0030@auburn.edu (M. Islam); sealscd@auburn.edu (C. Seals)



[1]https://github.com/leonidasengineer-star/touche-clef-seals-nlp

embeddings, as in the Qwen3 family [8]. For the encoders, we evaluate BERT [9], RoBERTa [10], and ModernBERT [11].

**Multi-task and contrastive supervision.** Multi-task learning improves generalization by sharing representations across related tasks [12]; we adopt a joint model that uses type and scheme classification for auxiliary supervision. For LLMs, whose next-token pretraining can favor shallow token-fitting over logical structure, latent-space contrastive frameworks [13] instead optimize the embedding manifold directly: supervised contrastive losses [3] cluster structurally sound reasoning while repelling fallacious patterns, sharpening class separation.

# 3. System Overview

## 3.1. Approaches and Architectures

We address all three subtasks simultaneously using two modeling approaches spanning diverse sizes and pretraining strategies, as outlined in Table 1:

- **Transformer Encoders:** We fully fine-tune bidirectional encoders, pooling token hidden states into a sequence representation routed to three linear classification heads for fallacy presence, type, and scheme.
- **Decoder LLMs:** We evaluate three adaptation depths on the Qwen3 family [14, 8] while freezing base weights to safeguard linguistic foundations: first, *frozen probes* using last-token embeddings from Qwen3-Embedding models with per-task logistic-regression probes; second, *zero-shot prompting* generating all three labels as JSON in a single pass; and third, *contrastive LoRA* [4] adapters on the attention projections, followed by the same linear probes.

Each configuration is trained and evaluated independently on both textual views, specifically the raw extracted text, denoted as TEXT_BASE, and the rhetorically enhanced form, denoted as TEXT_ENHANCED, isolating enrichment gains from architectural effects.

**Table 1**
Pretrained architectures and tuning approaches evaluated in this work.

| Model Type | Architecture | Parameters | Strategy |
|---|---|---|---|
| Transformer Encoders | `bert-base` | 110M | Full Fine-Tuning |
| | `roberta-base` | 125M | |
| | `roberta-large` | 355M | |
| | `ModernBERT-base` | 149M | |
| | `ModernBERT-large` | 395M | |
| Frozen Embedders | `Qwen3-Embedding-0.6B` | 0.6B | Linear Probe |
| | `Qwen3-Embedding-4B` | 4B | |
| | `Qwen3-Embedding-8B` | 8B | |
| Contrastive LLMs | `Qwen3-Embedding-0.6B` | 0.6B | SupCon LoRA + Probe |
| | `Qwen3-0.6B` (base LM) | 0.6B | |
| Zero-Shot LLMs | `Qwen3-0.6B` | 0.6B | Prompting only |
| | `Qwen3-4B-Instruct` | 4B | |
| | `Qwen3-8B` | 8B | |

## 3.2. Masked Multi-Task and Contrastive Losses

For the fully fine-tuned encoders, the three classification heads are optimized jointly using masked cross-entropy. The binary head ($\mathcal{L}_{T1}$) is trained on all instances, the type head ($\mathcal{L}_{T2}$) only on fallacious

instances, and the scheme head ($\mathcal{L}_{T3}$) only on non-fallacious instances. The total multi-task loss is a weighted sum: $\mathcal{L}_{encoder} = \mathcal{L}_{T1} + \mathcal{L}_{T2} + \lambda\mathcal{L}_{T3}$, where $\lambda = 0.5$ counterbalances the skewed class distribution within the scheme subset.

For the LLM adapters, we directly optimize the embedding manifold using a supervised contrastive (SupCon) objective [3]. Since every non-fallacious instance records the fallacy it most resembles, we define 16 contrastive classes: 8 true fallacies and 8 factual counterparts that resemble them. For a batch of embeddings $z$ in a multi-view setup, the loss is formulated as:

$$\mathcal{L}_{supcon} = \sum_{i \in I} \frac{-1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp(z_i \cdot z_p / \tau)}{\sum_{a \in A(i)} \exp(z_i \cdot z_a / \tau)} \tag{1}$$

where $I$ is the set of batch indices, and $z$ denotes an embedding. Specifically, $z_i$, $z_p$, and $z_a$ are the embeddings for the anchor index $i$, a positive pair index $p$, and an arbitrary batch index $a$, respectively. Here, $A(i)$ is the set of all other indices in the batch, $\tau = 0.07$ is a scalar temperature following the empirical defaults of Khosla et al. [3], and $P(i)$ is the set of positive pairs that share the exact same contrastive class as the anchor $i$. This explicitly forces the model to geometrically separate fallacious structures from their closely related, factual counterparts.

# 4. Experimental Setup

## 4.1. Data

We use the official Touché 2026 Fallacy Detection training set of 938 arguments and test set of 233 arguments [1, 2], both split nearly evenly into fallacious and non-fallacious subsets, as detailed in Table 2. Fallacious arguments carry one of 8 fallacy types, distributed near-uniformly at 10 to 14% each. Non-fallacious arguments carry an argument scheme, which is the cross-product of argument *goal* and *basis*, whose distribution is heavily skewed: *epistemic-internal* covers 68% of instances, while *practical-external* has 8 training instances and none in the test set. We use no external data or augmentation.

**Table 2**
Class distributions in the training set of 938 samples and the test set of 233 samples. Fallacy types are well-balanced; the scheme distribution is skewed, with *practical-external* absent from the test set.

| **Fallacious (Subtask 2)** | | | **Non-Fallacious (Subtask 3)** | | |
|---|---|---|---|---|---|
| Fallacy Type | Train | Test | Argument Scheme | Train | Test |
| Appeal to Tradition | 65 | 16 | Epistemic-Internal | 322 | 94 |
| Slippery Slope | 62 | 15 | Practical-Internal | 93 | 19 |
| Appeal to Nature | 61 | 15 | Epistemic-External | 50 | 5 |
| Relative Privation | 61 | 15 | Practical-External | 8 | 0 |
| Appeal to Authority | 59 | 15 | | | |
| Hasty Generalization | 55 | 14 | | | |
| False Dilemma | 54 | 13 | | | |
| Appeal to Population | 48 | 12 | | | |
| **Total** | **465** | **115** | **Total** | **473** | **118** |

## 4.2. Training Configuration

Inputs are truncated to 256 tokens, covering more than 95% of both text variants, with a median length of 59 tokens for base text and 116 for enhanced text. Encoders undergo full fine-tuning, the contrastive Qwen3 models train rank-16 LoRA adapters on their attention projections, and zero-shot variants only use a single pass. The full per-architecture training grid is deferred to Appendix A, and the zero-shot prompt to Figure B1.

### 4.3. Evaluation

Each trainable method is trained on 100% of the training data and scored in a single pass on the released test set. We report macro-$F_1$ over the fixed official label inventory, with unparseable generations falling back to a default class: the binary head, referred to as T1, is scored on all instances, the type head, or T2, on the gold-fallacious subset, and the scheme head, or T3, on the gold-non-fallacious subset.

## 5. Results and Discussion

### 5.1. Main Results

Table 3 reports macro-$F_1$ for all systems on the released test set, for both input variants. The most noticeable effect is input enrichment: when latent premises are spelled out, every system improves dramatically. This indicates that the bottleneck is not model capacity but the highly implicit nature of raw argumentative text. We therefore regard the base text as the scientifically informative test bed, and there, fully fine-tuned encoders dominate: every encoder beats every LLM configuration on T1, with ModernBERT-large leading detection, and RoBERTa-large leading scheme classification. This suggests a division of labor: leveraging the reasoning of larger models to first make the fallacious argumentation explicit, which small fine-tuned detectors then classify more accurately. Contrastive adaptation proves a targeted tool rather than a general win: relative to the frozen 0.6B probe, SupCon LoRA lifts type and scheme classification, but *degrades* the binary boundary of T1, as clustering by fine-grained class competes with the coarse fallacy separation. The plain Qwen3-0.6B base LM reaches similar results, so this geometry is created by the contrastive objective rather than the model. Finally, the enhanced RoBERTa-large run, our official submission, exactly reproduces its Subtask 1 leaderboard score of 0.957, as shown in Table 4, confirming our local pipeline replicates the graded submission.

**Table 3**
Test-set macro-$F_1$ using full training data. Best in bold. T1: Fallacy Detection. T2: Fallacy Classification. T3: Scheme Classification.

| | Base | | | Enhanced | | |
|---|---|---|---|---|---|---|
| Model Configuration | T1 | T2 | T3 | T1 | T2 | T3 |
| *Fully Tuned Encoders* | | | | | | |
| BERT-base | 0.704 | 0.552 | 0.375 | 0.923 | 0.937 | **0.729** |
| RoBERTa-base | 0.699 | 0.667 | 0.374 | **0.957** | 0.937 | 0.721 |
| RoBERTa-large | 0.764 | 0.703 | **0.402** | **0.957** | 0.956 | 0.671 |
| ModernBERT-base | 0.716 | 0.576 | 0.331 | 0.940 | 0.954 | 0.718 |
| ModernBERT-large | **0.767** | **0.794** | 0.336 | **0.957** | **0.974** | 0.727 |
| *Frozen Embedder + Linear Probe* | | | | | | |
| Qwen3-Embed-0.6B | 0.678 | 0.566 | 0.248 | 0.738 | 0.844 | 0.348 |
| Qwen3-Embed-4B | 0.661 | 0.585 | 0.245 | 0.781 | 0.920 | 0.439 |
| Qwen3-Embed-8B | 0.695 | 0.629 | 0.248 | 0.798 | 0.882 | 0.483 |
| *Contrastive LoRA + Linear Probe* | | | | | | |
| Qwen3-Embed-0.6B | 0.643 | 0.671 | 0.330 | 0.909 | 0.953 | 0.718 |
| Qwen3-0.6B base LM | 0.639 | 0.533 | 0.329 | 0.918 | 0.953 | 0.595 |
| *Zero-Shot LLMs* | | | | | | |
| Qwen3-0.6B | 0.472 | 0.120 | 0.171 | 0.509 | 0.271 | 0.246 |
| Qwen3-4B-Instruct | 0.624 | 0.553 | 0.168 | 0.901 | 0.955 | 0.355 |
| Qwen3-8B | 0.647 | 0.597 | 0.221 | 0.787 | 0.946 | 0.233 |

### 5.2. Official Submission and Leaderboard Results

We submitted one run per subtask via TIRA [15] under the team name SEALS-NLP as detailed in Table 4.

**Table 4**
Top-5 official Touché 2026 leaderboard runs sorted by macro-$F_1$. Our enhanced-track submission, Seals-NLP, is in bold.

| | Base track | | Enhanced track | |
|---|---|---|---|---|
| Rank | Team | $F_1$ | Team | $F_1$ |
| *Subtask 1 (Fallacy Detection)* | | | | |
| 1 | grtsh | 0.807 | **seals-nlp (ours)** | **0.957** |
| 2 | uzhcl-for-the-win | 0.751 | uzhcl-for-the-win | 0.944 |
| 3 | veritas | 0.737 | ooooooo | 0.940 |
| 4 | helium | 0.736 | veritas | 0.927 |
| 5 | helium | 0.736 | sevitha | 0.927 |
| *Subtask 2 (Fallacy Classification)* | | | | |
| 1 | uzhcl-for-the-win | 0.859 | uzh-tt | 0.991 |
| 2 | grtsh | 0.671 | uzh-tt | 0.991 |
| 3 | grtsh | 0.638 | clef26-sophisma | 0.972 |
| 4 | organizer | 0.050 | **seals-nlp (ours)** | **0.963** |
| 5 | | | sevitha | 0.956 |
| *Subtask 3 (Scheme Classification)* | | | | |
| 1 | uzhcl-for-the-win | 0.716 | sevitha | 0.976 |
| 2 | grtsh | 0.700 | sevitha | 0.880 |
| 3 | organizer | 0.498 | uzh-tt | 0.858 |
| 4 | | | **seals-nlp (ours)** | **0.816** |
| 5 | | | uzhcl-for-the-win | 0.748 |

## 6. Limitations

All results are single training runs on a small test set, so minor score differences should not be over-interpreted, and the lack of *practical-external* scheme test examples caps the Subtask 3 scores near 0.75. Our LLM experiments are confined to the Qwen3 family at up to 8B parameters with a single prompt template, and the contrastive hyperparameters were set a priori rather than tuned. Finally, since the enhanced text explicitly articulates each argument's latent reasoning, it partially encodes the annotation itself. The near-saturated scores across all architectures therefore likely overstate what is attainable on naturally occurring arguments.

## 7. Conclusion

We presented the Seals-NLP system for the Touché 2026 Fallacy Detection challenge. Our multi-task RoBERTa-large, fine-tuned on the enhanced text, ranked first on the enhanced track of Subtask 1 in the official leaderboard. On raw text, fully fine-tuned encoders remained superior to prompted and adapted LLMs at every scale that we tested, while contrastive adaptation aided fine-grained classification at the cost of binary fallacy detection. Together, these results indicate that the central difficulty of fallacy detection is recovering implicit premises rather than classifying explicit ones. This is further supported by our exploratory experiments with argument scaffolds, distillation, and cascading tasks (Appendix C). Future work will target this bottleneck by using advanced reasoning LLMs to dynamically reconstruct implicit premises, optimizing the generated text for downstream detector accuracy.

## Acknowledgments

We thank the Touché 2026 organizers for releasing the dataset, TIRA [15] for its submission infrastructure, and the Auburn University CSSE department for computational resources.

## Declaration on Generative AI

During the preparation of this work, the authors used Google Gemini to draft content, improve writing style, perform grammar and spelling checks, and assist with formatting. Following the use of this tool, the authors reviewed and edited the content as needed, and take full responsibility for the publication's final text.

## References


[1] J. Kiesel, M. Feger, T. Hagen, S. Heineking, M. Heinrich, M. Fröbe, K. Boland, C. Mathews, W. Pertsch, J. Romberg, I. Zelch, S. Dietze, M. Hagen, M. Potthast, B. Stein, Overview of touché 2026 — argumentation systems, in: M. Hagen, M. Potthast, B. Stein, P. Schaer, E. Zangerle, S. MacAvaney, J. M. Struß, E. S. Salido, A. Barrón-Cedeño, A. G. S. Herrera (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction. 17th International Conference of the CLEF Association (CLEF 2026), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2026.

[2] J. Kiesel, M. Feger, T. Hagen, S. Heineking, M. Heinrich, M. Fröbe, K. Boland, C. Mathews, W. Pertsch, J. Romberg, I. Zelch, S. Dietze, M. Hagen, M. Potthast, B. Stein, Overview of touché 2026 — argumentation systems — extended version, in: E. S. Salido, A. Barrón-Cedeño, A. G. S. Herrera, S. MacAvaney, J. M. Struß (Eds.), CLEF 2026 Working Notes, CEUR Workshop Proceedings, CEUR-WS.org, 2026.

[3] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, D. Krishnan, Supervised contrastive learning, in: Advances in Neural Information Processing Systems 33 (NeurIPS 2020), Curran Associates, Inc., 2020, pp. 18661–18673. URL: https://proceedings.neurips.cc/paper/2020/hash/d89a66c7c80a29b1bdbab0f2a1a94af8-Abstract.html.

[4] 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: Proceedings of the 10th International Conference on Learning Representations (ICLR), OpenReview.net, 2022. URL: https://openreview.net/forum?id=nZeVKeeFYf9.

[5] Z. Jin, A. Lalwani, T. Vaidhya, X. Shen, Y. Ding, Z. Lyu, M. Sachan, R. Mihalcea, B. Schölkopf, Logical fallacy detection, in: Findings of the Association for Computational Linguistics: EMNLP 2022, Association for Computational Linguistics, 2022, pp. 7180–7198. doi:10.18653/v1/2022.findings-emnlp.532.

[6] A. Ramponi, A. Daffara, S. Tonelli, Fine-grained fallacy detection with human label variation, in: Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), Association for Computational Linguistics, 2025, pp. 762–784. doi:10.18653/v1/2025.naacl-long.34.

[7] J. Kiesel, Ç. Çöltekin, M. Gohsen, S. Heineking, M. Heinrich, M. Fröbe, T. Hagen, M. Aliannejadi, S. Anand, T. Erjavec, M. Hagen, M. Kopp, N. Ljubešić, K. Meden, N. Mirzakhmedova, V. Morkevičius, H. Scells, M. Wolter, I. Zelch, M. Potthast, B. Stein, Overview of Touché 2025: Argumentation Systems, in: J. C. 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. 16th International Conference of the CLEF Association (CLEF 2025), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2025. doi:10.1007/978-3-031-88720-8_67.

[8] Y. Zhang, M. Li, D. Long, X. Zhang, H. Lin, B. Yang, P. Xie, A. Yang, D. Liu, J. Lin, F. Huang, J. Zhou, Qwen3 embedding: Advancing text embedding and reranking through foundation models, 2025. arXiv:2506.05176.

[9] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long

and Short Papers), Association for Computational Linguistics, 2019, pp. 4171–4186. doi:10.18653/v1/N19-1423.

[10] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, RoBERTa: A robustly optimized BERT pretraining approach, 2019. arXiv:1907.11692.

[11] B. Warner, A. Chaffin, B. Clavié, O. Weller, O. Hallström, S. Taghadouini, A. Gallagher, R. Biswas, F. Ladhak, T. Aarsen, G. T. Adams, J. Howard, I. Poli, Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference, in: Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, 2025, pp. 2526–2547. doi:10.18653/v1/2025.acl-long.127.

[12] M. Crawshaw, Multi-task learning with deep neural networks: A survey, 2020. arXiv:2009.09796.

[13] L. Shan, H. Chen, Y. Wang, Z. Liu, W. Li, Latent-space contrastive reinforcement learning for stable and efficient LLM reasoning, 2026. arXiv:2601.17275.

[14] A. Yang, A. Li, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Gao, C. Huang, C. Lv, et al., Qwen3 technical report, 2025. arXiv:2505.09388.

[15] M. Fröbe, M. Wiegmann, N. Kolyada, B. Grahm, T. Elstner, F. Loebe, M. Hagen, B. Stein, M. Potthast, Continuous integration for reproducible shared tasks with tira.io, in: J. Kamps, L. Goeuriot, F. Crestani, M. Maistro, H. Joho, B. Davis, C. Gurrin, U. Kruschwitz, A. Caputo (Eds.), Advances in Information Retrieval. 45th European Conference on IR Research (ECIR 2023), Lecture Notes in Computer Science, Springer, Berlin Heidelberg New York, 2023, pp. 236–241. doi:10.1007/978-3-031-28241-6_20.

[16] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, D. Zhou, Chain-of-thought prompting elicits reasoning in large language models, in: Advances in Neural Information Processing Systems 35 (NeurIPS 2022), 2022. URL: https://proceedings.neurips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.

[17] V. Vapnik, R. Izmailov, Learning using privileged information: Similarity control and knowledge transfer, Journal of Machine Learning Research 16 (2015) 2023–2049. URL: http://jmlr.org/papers/v16/vapnik15b.html.

[18] G. Hinton, O. Vinyals, J. Dean, Distilling the knowledge in a neural network, 2015. arXiv:1503.02531.

[19] J. Read, B. Pfahringer, G. Holmes, E. Frank, Classifier chains for multi-label classification, Machine Learning 85 (2011) 333–359. doi:10.1007/s10994-011-5256-5.

# A. Hyperparameters

Table A1 reports the core training configurations. All runs use AdamW, weight decay 0.01, 10% linear warmup, random seed 42, and a single 32 GB NVIDIA GeForce RTX 5090.

**Table A1**
Training configurations.

| Setting | Value |
|---|---|
| *Encoder fine-tuning* | |
| Learning rate | $5 \times 10^{-5}$ (RoBERTa-large: $1 \times 10^{-5}$) |
| Epochs | 6 (RoBERTa-large: 10) |
| Per-device batch size | 8 |
| Pooling strategy | Mean-pooling (BERT-base: CLS token) |
| Maximum sequence length | 256 tokens (dynamic padding) |
| *Contrastive LoRA (SupCon)* | |
| Adapter | LoRA, $r$=16, $\alpha$=32, dropout 0.05, on q, k, v, o |
| Learning rate | $2 \times 10^{-4}$ |
| Epochs | 16 |
| Batch size | 64 |
| Temperature $\tau$ | 0.07 |
| Contrastive classes | 16 (8 fallacy types + 8 resembles) |
| Scheme (T3) term weight | 0.5 |
| Pooling strategy | Last-token |
| Probe | Logistic regression, $C$=1.0 |
| Maximum sequence length | 256 tokens (train), 512 (inference) |

# B. Zero-Shot Prompt

The zero-shot runs elicit all three subtask labels in a single JSON-constrained pass, with the simplified system prompt shown in Figure B1.

```
You are an expert annotator determining whether fallacies exist in an argument (where
    1=fallacious and 0=factual).
Given a comment, decide exactly one option for each of the following categories:
1. fallacy_exists: [1, 0].
2. fallacy_type: ["authority", "black-white", "hasty_generalization", "natural", "population",
    "slippery_slope", "tradition", "worse_problems"].
3. argument_goal: ["practical", "epistemic"].
4. argument_basis: ["internal", "external"].
Respond ONLY in the following JSON object example structure:
{"fallacy_exists": 1, "fallacy_type": "authority", "argument_goal": "epistemic",
    "argument_basis": "internal"}

Comment:
{comment_text}
```

**Figure B1:** System prompt used for zero-shot LLM evaluations.

# C. Explorations: Scaffolds, Distillation, and Coupling

We ran additional experiments on our base model (ModernBERT-large) to see if we could bridge the performance gap between the base and enhanced tracks without relying on extra data during testing (Table C1).

**Table C1**
Exploratory test-set experiments: single runs on ModernBERT-large exploring argument scaffolds, privileged information distillation from the enhanced view, and cascading tasks. Best per column in bold; first row = the baseline from Table 3.

| Configuration | T1 | T2 | T3 |
|---|---|---|---|
| ModernBERT-large baseline (Table 3) | 0.767 | 0.794 | 0.336 |
| *Argument scaffolds* | | | |
| + ARGUMENT_BASE | **0.789** | **0.867** | 0.434 |
| + Qwen3-4B generated | 0.720 | 0.750 | **0.442** |
| *Privileged information distillation* | | | |
| + distillation (T1 head only) | 0.785 | 0.742 | 0.405 |
| + distillation (all three heads) | 0.755 | 0.787 | 0.371 |
| + scaffold + distillation (T1 head) | 0.785 | 0.806 | 0.419 |
| *Cascading tasks* | | | |
| cascade: T1 $\rightarrow$ T2, T3 | 0.785 | 0.746 | 0.397 |
| cascade: T2, T3 $\rightarrow$ T1 | 0.757 | 0.754 | 0.369 |
| heuristic: T3 > T2 confidence $\rightarrow$ T1 | 0.733 | — | — |

**Argument scaffolds.** Every instance includes a breakdown of its argument structure from ARGUMENT_BASE. Simply adding this breakdown to the input, much like a chain-of-thought reasoning trace [16], improves performance across all tasks, giving us our best base-track scores (top section of Table C1). This suggests that clearly spelling out the argument helps the model spot fallacies. However, if we replace this perfect, human-made breakdown with one generated by Qwen3-4B, performance drops significantly. This shows that the quality of the scaffold is critical to the model's success.

**Privileged information distillation.** We also explored utilizing extra data strictly during training, known as Learning Using Privileged Information (LUPI) [17]. This technique leverages the enhanced text during training even though it remains unavailable at test time. Specifically, we trained a "teacher" model on the rich enhanced text and used it to guide a "student" model that only has access to the base text [18]. As shown in the middle section of the table, this approach was effective: distilling the single binary task allowed the student model to nearly match the performance gains of the argument scaffold without requiring any extra inputs during testing. However, the benefits are strictly limited to the primary fallacy detection task; attempting to distill all three tasks simultaneously actually worsened its performance. Furthermore, combining this distillation method with the argument scaffold yielded no additional improvements, which suggests that both techniques are ultimately helping the model learn the same underlying patterns.

**Cascading tasks.** Another interesting approach is using one prediction task to determine the outcome of another. For example, the bottom section of the table shows that simply guessing an argument is fallacious whenever the model is more confident about its specific fallacy type than its general scheme works surprisingly well. We can also cascade these tasks directly, in the spirit of classifier chains [19]. Feeding the simple binary fallacy prediction forward into the detailed type predictions improves the overall score, though it is not quite as effective as using the argument scaffold. Finally, reversing the cascade by using detailed predictions to help the basic fallacy guess makes performance noticeably worse.

**Exploratory findings.** We noticed a clear pattern across these three different experiments: trying to force detailed category information directly into the simple binary fallacy detection always degrades performance, whereas indirect connections help. Interestingly, all three of our very different methods hit a similar performance ceiling. This suggests a fundamental gap between small detection models and larger reasoning models that training techniques alone cannot bridge.