StructICL: Enhancing Few-Shot Logical Reasoning via Structure-Preserving Contrastive Learning

Anonymous ACL submission

Abstract

Few-shot learning is a viable alternative to fine-tuning under data scarcity or when using proprietary black-box language models. We propose a simple contrastive framework for in-context example selection by learning a task-specific similarity metric. We evaluate our approach on Logical Reasoning Evaluation (LRE) and show consistent improvements over untrained retrievers and random selection. Averaged across five 4B and 8B models, our approach yields a 2% accuracy gain over the baseline. This improvement remains robust across four LRE datasets and two retriever models. We will make our code and data publicly available in the final version of the paper.

1 Introduction

In the era of large black-box large language models (LLMs), often combined in complex agentic pipelines, few-shot learning is one of the most realistic techniques to improve model performance. Also dubbed as in-context learning (ICL), it provides task-specific knowledge by adding several demonstration samples to the model input. The role of such examples is two-fold: they both specify the target output format and enhance task understanding.

However, relatively little attention is paid to the algorithm of example selection. The authors often use the same selection of random samples (Min et al., 2022) for any test example or rely on pretrained embedders for retrieving similar demonstrations (Margatina et al., 2023; Nori et al., 2023). We hypothesize that general semantic simlarity is suboptimal for several tasks and might be outperformed by task-specific embeddings. Naively speaking, when solving a math or coding problem, the more useful hints are not the problems that share the same variable naming but the ones that require similar algorithms or techniques.

Our work is inspired by the one of Sorokin and Nasyrova (2025). They apply ICL to LLM-based grammatical error correction and induce domain similarity by using an encoder that predicts edit types. They further enhance this encoder by training it to assign similar representations to examples corresponding to the same grammar rules. However, as their approach heavily relies on the fine-grained label structure and the rule type annotation, it does not readily generalize to arbitrary problems.

We propose to fill this gap by training a task-specific similarity model via contrastive learning. We explicitly prompt an LLM to create an analogous example for each item in the training data. Then, we use these pairs of original samples and their generated augmented views to finetune an embedding model. By design, this model assigns similar vector representations to structurally analogous examples and can be utilized to retrieve the most useful demonstrations.

We apply our method to the problem of logical reasoning evaluation (LRE, Qi et al. (2025); Han et al. (2024)). This task naturally decouples surface verbose representations from underlying logical principles. Additionally, in contrast to mathematics and coding, logical reasoning data is not abundant in LLM pretraining data, thus leaving more room to improve model abilities via demonstration selection. We verify two claims:

1. Medium-sized language models benefit from similar example selection in the task of logical reasoning evaluation.
2. Task-specific similarity might be learned using synthetic pairwise data.

Though we limit our research to logical reasoning, our approach might be applied to any NLP task where the notion of structurally similar example makes sense. To the best of our knowledge, we are the first to apply this straightforward yet effective method to few-shot learning.

2 Methodology

2.1 Logical reasoning evaluation

The task of logical reasoning evaluation is usually framed as a classification task and consists of predicting whether a statement is true, false, or undefined given a set of premises. In contrast to the well-known task of natural language inference (NLI), it focuses not on commonsense reasoning but on strict deduction formalized via first-order logic (FOL). Usually, the premise includes several declarative statements (*Abigail is tricky*, *Both Marc and Abigail are not green*) and inference rules (*If someone is not green, he/she is either tricky or curious, but not both*), while the conclusion consists of a single statement (*Abigail is not curious*).

The predicate names and inference rules are intentionally meaningless to ensure the model relies only on formal logical laws, not on general knowledge or common sense. For most existing datasets the instances also include corresponding formal FOL statements for the premises and conclusion. For a more complete exposition to the problem refer to Qi et al. (2025).

2.2 Retrieval-based in-context learning

When applied to a LRE instance $x = (p, c)$, a large language model M together with a prompt p defines a function $f_{M,p}$ producing an output $f_{M,p}(x) = y \in \{\text{True}, \text{False}, \text{Uncertain}\}$. In the few-shot mode the input is enhanced with K additional samples (demonstrations) $D_K(x) = \{x^1, \ldots, x^K\}$ together with their labels $y^1, \ldots, y^K$:

$$y = f_{M,p}(x, [(x^1, y^1), \ldots, (x^K, y^K)]).$$

These demonstrations are selected from an annotated training set $T = [(x_1, y_1), \ldots, (x_N, y_N)]$. Throughout the paper we assume that $D_K(x)$ includes exactly the K nearest neighbours of x in T according to some distance measure d:

$$D_K(x) = \text{Close}_{K,d}(x, T).$$

An embedder E returns a vector representation h for each LRE instance $x = (p, c)$: $h_{E,x} = E(x)$. Every embedder induces a distance measure for input instances: $d_E(x, x') = \rho(h_{E,x}, h_{E,x'})$, where ρ is a metric (e.g., euclidean or cosine). The primary goal of this study is to develop an embedding model E that selects the most beneficial demonstration examples to maximize few-shot prediction accuracy for the LRE task.

2.3 Self-supervised contrastive learning

In our study we follow the self-supervised contrastive learning approach (Chen et al., 2020), pulling together the vector representations of the baseline instance and its augmented version. We enforce the augmented sample to satisfy two main principles:

1. The original example and its augmentation share the same underlying logical reasoning structure to reach the solution.
2. However, they use different phrasing (e.g., variable and predicate names) and thus have low surface similarity.

We explicitly formulate these requirements in the LLM prompt[1] (Fig. 1). On the FOLIO training set we verify that this prompt actually produces fully isomorphic samples in 95.4% of cases. Applying the augmentation prompt to the training data, we obtain a dataset of positive pairs that is further used to tune a pretrained text embedder using contrastive learning. The training process is described in Subsection 3.2.

3 Experiments and results

3.1 Datasets

We select four datasets for our main evaluation: FOLIO (Han et al., 2024), ProofWriter (Tafjord et al., 2021), ProverQA (Qi et al., 2025) and LogicNLI (Tian et al., 2021). Their parameters are provided in Table 1. The description of datasets and their splits is provided in Appendix D.

Dataset	Labels	Train	Val	Test
FOLIO	3	900	101	203
ProofWriter	3	8,046	982	976
ProverQA	3	4,000	1,000	1,500
LogicNLI	4	16,000	2,000	2,000

Table 1: Dataset parameters used for evaluation.

3.2 Embedder training

We finetune the embedder on the training splits of our data, using pairs $(x_i, \tilde{x}_i)$, where x_i is a training instance and $\tilde{x}_i$ its analogical counterpart generated as described in Section 2.3, using contrastive learning objective. InfoNCE loss (Oord et al., 2018) is used for training, the loss is further wrapped

[1] We refrain from using rule-based approach to create isomorphic examples as it does not generalize well to other tasks.

```
Dear Model, you will be given a logical reasoning problem with premises and a conclusion.
The conclusion can be True, False, or Uncertain based on the premises.
Your task: Generate 1 similar problem that preserves the EXACT logical structure and
reasoning pattern, but uses completely different domain/context.

Key requirements:
- Maintain the same logical relationships between entities
- Preserve any irrelevant information if present in original premises
- Keep the same conclusion validity (True/False/Uncertain)
- Use different semantic domains (e.g., if original is about animals, use
technology/sports/etc.)

Format: Return only JSON with "Premises" (list), "Conclusion" (string), "Answer"
(True/False/Uncertain).

Original problem:
Premises: {premises}
Conclusion: {conclusion}
Answer: {answer}
```

Figure 1: Duplication prompt for the FOLIO dataset. For other datasets class names are changed accordingly.

with MatryoshkaLoss (Kusupati et al., 2022). More details on embedder training are provided in Appendix B.

3.3 Evaluation

Our main claim is the superiority of finetuned embedder (FT-embedder) over the pretrained one (PT-embedder). Therefore we compare several in-context learning setups:

1. Zero-shot learning.
2. Few-shot learning with 3 random examples.
3. Few-shot learning with 3 most similar examples using pretrained embedder.
4. Few-shot learning with 3 most similar examples using finetuned embedder (our proposal).
5. Full finetuning (skyline).

We use Qwen3-Embedding-0.6B[2] (Zhang et al., 2025a) as the backbone embedder and evaluate three variants of Qwen3 models: Qwen3-4B-Instruct[3], Qwen3-4B-Thinking[4] and Qwen3-8B[5] and two variants of Phi-4-mini models: Phi-4-mini-instruct[6] (Abouelenin et al., 2025) and Phi-4-mini-reasoning [7] (Xu et al., 2025). As the distribution of labels is almost uniform (see Table 5), accuracy is used as main evaluation metric.

Results in Table 2 show that the proposed FT-embedder consistently outperforms both the pretrained one and the random few-shot examples demonstrating the best performance for 16 out of 20 model-dataset combinations. The advantage is more pronounced for the smaller 4B models and for the smallest dataset (FOLIO) containing only 900 training examples where the FT-embedder even beats full finetuning.

4 Ablation studies

All ablations are performed on FOLIO dataset with Qwen-3 models.

4.1 Additional experiments

First, we ask whether benefits from constrastive embedder tuning are bounded to a particular embedding type or hold for other models as well? Since Qwen3-Embedding models are instructive and decoder-based, for comparison we select encoder-based `e5-base-v2`[8] (Wang et al., 2022). The results, reported in Table 7, show accuracy gain of 2% and confirm that contrastive tuning success generalizes across embedder architectures and scale.

Second, we ask whether contrastive embedder tuning is still beneficial for larger number of few-shot samples. Figure 7 shows that FT-embedder continues to outperform PT-embedder for $k = 5, 10$ and 4B models, while for a larger Qwen3-8B

[2] https://huggingface.co/Qwen/Qwen3-Embedding-0.6B
[3] https://huggingface.co/Qwen/Qwen3-4B-Instruct-2507
[4] https://huggingface.co/Qwen/Qwen3-4B-Thinking-2507
[5] https://huggingface.co/Qwen/Qwen3-8B
[6] https://huggingface.co/microsoft/Phi-4-mini-instruct
[7] https://huggingface.co/microsoft/Phi-4-mini-reasoning
[8] https://huggingface.co/intfloat/e5-base-v2

	4B-INSTRUCT				4B-THINKING				8B				PHI-4-INSTRUCT				PHI-4-REASONING			
Method	F	LN	PW	PQ	F	LN	PW	PQ	F	LN	PW	PQ	F	LN	PW	PQ	F	LN	PW	PQ
Zero	62.0	40.0	64.1	61.3	68.0	36.4	65.0	67.9	77.8	48.9	86.3	80.9	67.5	41.6	62.0	62.8	67.5	45.2	53.3	59.2
Random	70.4	38.1	**79.3**	69.3	75.9	36.3	76.8	75.9	83.7	49.0	**89.5**	82.7	65.0	43.0	62.8	61.5	73.9	41.4	59.9	62.4
PT	72.9	43.3	77.0	67.8	80.3	38.0	79.8	75.6	85.2	50.2	88.4	83.0	70.9	42.5	61.6	62.8	70.9	41.8	57.9	60.7
FT	**80.8**	**43.7**	78.2	**71.2**	**85.7**	**39.3**	**82.7**	**78.0**	**88.2**	**51.5**	*87.8*	**83.7**	*69.0*	**46.8**	**63.4**	*62.8*	**75.4**	**42.0**	**61.3**	**63.5**
Full FT	77.3	77.3	92.3	87.8	81.3	77.3	94.0	89.2	66.5	69.7	92.9	88.8	70.4	68.9	81.6	83.2	69.0	62.8	84.8	84.8

Table 2: Accuracy (%) across datasets and models (F=FOLIO, LN=LogicNLI, PW=ProofWriter, PQ=ProverQA). **Bold**: best among few-shot methods (rows 2–4) per column. *Italics*: FT embedder doesn't exceed PT embedder. Underline: finetuned model underperforms best few-shot configuration.

model all selection strategies show similar performance. We consider several other modifications of embedder tuning pipeline in Appendix E.

4.2 Improvements analysis

To assess whether contrastive training helps retrieve examples with similar reasoning paths, we compared demonstrations from the fine-tuned (FT) and pre-trained (PT) embedders relative to the anchor example. Using the evaluation prompt shown in Figure 6, the FT-embedder achieved a $69/31$ win rate over the PT baseline, supporting its ability to retrieve more structurally similar samples.

To evaluate quantitatively how the FT-embedder improves few-shot example selection, we evaluate the accuracy of corresponding kNN classifier. As shown in Table 3, PT-embedder retrieval accuracy is only slightly above chance (33.3%). Contrastive tuning enhances retrieval quality up to $\sim 48\%$ in case of 1 nearest neighbour and to 50% and 53% in case of 3 neighbours. Thus, the geometry of the finetuned vector representations better aligns with the underlying structure of logical reasoning.

Retrieval method	**kNN-1**	**kNN-3**
`e5-base-v2`	37.9	34.0
`Qwen3-Embedding-0.6B`	35.5	34.0
`e5-base-v2`**+FT**	47.8	50.7
`Qwen3-Embedding-0.6B`**+FT**	48.8	53.7

Table 3: Accuracy (%) of kNN classifiers using pretrained and contrastively finetuned (**+FT**) embedders as the retrieval backbone.

5 Related work

One of the first works to compare different principles of in-context example selection was Margatina et al. (2023) who demonstrated that similarity method outperforms other approaches, such as diversity and uncertainty, when applied to various classification and multi-choice tasks. The efficiency of this approach was demonstrated for various fields of NLP, such as machine translation (Zebaze et al., 2025), grammatical error correction (Sorokin and Nasyrova, 2025; Peng et al., 2025) and medical question answering (Nori et al., 2023). However, most of these works utilized pretrained embedders without their tuning for particular task. Some works used embedding not of raw texts, but of task-specific structured information, such as syntactic structures (Chakma et al., 2026) or error-correction pairs (Peng et al., 2025).

Embedder tuning for better retrieval of demonstrations goes back to the work of Rubin et al. (2022) who proposed to select demonstrations that improve probability of correct answers. Subsequent works also pursued this approach either by training a reward model (Wang et al., 2024), directly optimizing in-context learning performance via Bayesian framework (Zhang et al., 2025b) or applying ranking techniques (Ji et al., 2026).

6 Conclusion

We have shown that performance of medium-size language models in the task of logical reasoning evaluation depends on the quality of demonstrations. Better few-shot examples can be retrieved by training the embedder to match the internal structure of logical statements using contrastive learning. The results are stable across models and embedders. Since the distinction between latent and surface similarity holds true not only for LRE but also for many other formal scientific fields, we hope our approach will prove fruitful for other disciplines. We leave this extension for future work.

Limitations

While we hypothesize that contrastive embedding training on structural similarity data is not restricted to a particular task, the results on Logical Reasoning Evaluation might provide an overly optimistic picture. First, LRE with its clear separation between surface affinity and underlying logical isomorphism, is a perfect testbed for applying contrastive methods based on intrinsic similarity. This is further exacerbated by the fact that most LRE datasets are constructed by verbalizing formal schemes, at which level intrinsic similarity is targeted. For other tasks, the interaction between the surface and latent structure might be more confounded.

While a fair comparison with existing few-shot selection methods – which primarily optimize correct answer probability – is necessary, their current implementations (e.g., (Wang et al., 2024)) lack necessary architectural flexibility to support arbitrary language models and embedders.

7 Ethical considerations

This work focuses on the abstract domain of logical reasoning and does not raise immediate ethical concerns. However, because our methodology utilizes large language models (LLMs) to generate synthetic data, full control over their outputs cannot be guaranteed. We conducted manual verifications of the LLM-generated content, which revealed no harmful, biased, or misleading outputs. However, adapting this approach to sensitive applications should be done with caution.

References

Abdelrahman Abouelenin, Atabak Ashfaq, Adam Atkinson, Hany Awadalla, Nguyen Bach, Jianmin Bao, Alon Benhaim, Martin Cai, Vishrav Chaudhary, Congcong Chen, and 1 others. 2025. Phi-4-mini technical report: Compact yet powerful multimodal language models via mixture-of-loras. *arXiv preprint arXiv:2503.01743*.

Aunabil Chakma, Mihai Surdeanu, and Eduardo Blanco. 2026. Structured semantic information helps retrieve better examples for in-context learning in few-shot relation extraction. *arXiv preprint arXiv:2601.20803*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PmLR.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. 2021. Transformers as soft reasoners over language. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, IJCAI'20.

Georg Gottlob, Letizia Tanca, and 1 others. 1989. What you always wanted to know about datalog(and never dared to ask). *IEEE transactions on knowledge and data engineering*, 1(1):146–166.

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, Lucy Sun, Alexander Wardle-Solano, Hannah Szabó, Ekaterina Zubova, Matthew Burtell, Jonathan Fan, Yixin Liu, Brian Wong, Malcolm Sailor, and 16 others. 2024. FOLIO: Natural language reasoning with first-order logic. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22017–22031, Miami, Florida, USA. Association for Computational Linguistics.

Matthew Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. *arXiv preprint arXiv:1705.00652*.

Yuwen Ji, Luodan Zhang, Haoran Que, Lei Shi, Wang Chao, Yue Zhang, and 1 others. 2026. Learning to rank for in-context example retrieval. *Advances in Neural Information Processing Systems*, 38:141590–141617.

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain, and 1 others. 2022. Matryoshka representation learning. *Advances in Neural Information Processing Systems*, 35:30233–30249.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Katerina Margatina, Timo Schick, Nikolaos Aletras, and Jane Dwivedi-Yu. 2023. Active learning principles for in-context learning with large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5011–5034, Singapore. Association for Computational Linguistics.

William McCune. 2005. Release of prover9. In *Mile high conference on quasigroups, loops and nonassociative systems, Denver, Colorado*.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Harsha Nori, Yin Tat Lee, Sheng Zhang, Dean Carignan, Richard Edgar, Nicolo Fusi, Nicholas King, Jonathan Larson, Yuanzhi Li, Weishung Liu, and 1 others. 2023. Can generalist foundation models outcompete special-purpose tuning? case study in medicine. *arXiv preprint arXiv:2311.16452*.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Guangyue Peng, Wei Li, Wen Luo, and Houfeng Wang. 2025. Encode errors: Representational retrieval of in-context demonstrations for multilingual grammatical error correction. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 21166–21180, Vienna, Austria. Association for Computational Linguistics.

Chengwen Qi, Ren Ma, Bowen Li, Binyuan Hui, Jinwang Wu, Yuanjun Laili, Conghui He, and 1 others. 2025. Large language models meet symbolic provers for logical reasoning evaluation. In *International Conference on Learning Representations*, volume 2025, pages 54186–54205.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2655–2671, Seattle, United States. Association for Computational Linguistics.

Alexey Sorokin and Regina Nasyrova. 2025. LLMs in alliance with edit-based models: advancing in-context learning for grammatical error correction by specific example selection. In *Proceedings of the 20th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2025)*, pages 517–534, Vienna, Austria. Association for Computational Linguistics.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. 2021. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3621–3634, Online. Association for Computational Linguistics.

Jidong Tian, Yitian Li, Wenqing Chen, Liqiang Xiao, Hao He, and Yaohui Jin. 2021. Diagnosing the first-order logical reasoning ability through LogicNLI. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3738–3747, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

Liang Wang, Nan Yang, and Furu Wei. 2024. Learning to retrieve in-context examples for large language models. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1752–1767, St. Julian's, Malta. Association for Computational Linguistics.

Haoran Xu, Baolin Peng, Hany Awadalla, Dongdong Chen, Yen-Chun Chen, Mei Gao, Young Jin Kim, Yunsheng Li, Liliang Ren, Yelong Shen, and 1 others. 2025. Phi-4-mini-reasoning: Exploring the limits of small reasoning language models in math. *arXiv preprint arXiv:2504.21233*.

Armel Randy Zebaze, Benoît Sagot, and Rachel Bawden. 2025. In-context example selection via similarity search improves low-resource machine translation. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 1222–1252, Albuquerque, New Mexico. Association for Computational Linguistics.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, and 1 others. 2025a. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*.

Zheng Zhang, Shaocheng Lan, Lei Song, Jiang Bian, Yexin Li, and Kan Ren. 2025b. Learning to select in-context demonstration preferred by large language model. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 11345–11360, Vienna, Austria. Association for Computational Linguistics.

Appendix

A Prompts

```
You are a helpful assistant specialized in analyzing logical arguments and deductive reasoning.
Please determine whether a conclusion follows from given premises.
Think step by step, then give your final answer in the format: 'FINAL ANSWER: True', 'FINAL
ANSWER: False', or 'FINAL ANSWER: Uncertain'.
```

Figure 2: System prompt for inference.

```
Here are some examples of logical reasoning problems:

Example 1:
Premises: {premises_1}
Conclusion: {conclusion_1}
Correct answer: {label_1}

[... k examples ...]

Now, please analyze this case:
Given these premises:
{premises}
Is the following conclusion logically valid?
{conclusion}

Answer True, False, or Uncertain, then explain your reasoning step by step.
```

Figure 3: User prompt template for few-shot inference (FOLIO and ProverQA). For LogicNLI and ProofWriter label names are changed accordingly. For zero-shot, the examples block is omitted.

```
You are a helpful assistant specialized in analyzing logical arguments and deductive reasoning.
Please determine whether a conclusion follows from given premises.
Give your final answer in the format: 'FINAL ANSWER: True', 'FINAL ANSWER: False', or 'FINAL
ANSWER: Uncertain'.

Given these premises:
{premises}
Is the following conclusion logically valid?
{conclusion}

Answer True, False, or Uncertain.

FINAL ANSWER: {label}
```

Figure 4: Prompt template for supervised fine-tuning. The model is trained to predict only the completion after `FINAL ANSWER:` with completion-only loss. For reasoning models, a `<think>\n\n</think>` token is prepended before `FINAL ANSWER`.

```
Instruct: Given a set of premises and a conclusion, retrieve premise-conclusion pairs that
follow the same logical reasoning pattern.

Query: Premises: {premises} Conclusion: {conclusion}
```

Figure 5: Query prefix used with the pre-trained Qwen3-Embedding-0.6B embedder to establish a prompted baseline (PT+prompt).

```
You are an expert in logical reasoning. You will be given an anchor statement and two
candidate statements. All statements consist of a list of premises and a conclusion. You task
is to decide which of the two statement is more similar to the anchor statement if one
compares the reasoning path required to derive or refute their conclusion based on premises.

Your answer should be a dictionary in json format with keys "answer", "first_isomorphic",
"second_isomorphic", "explanation". "answer" field answers which of the statements is more
close to the anchor statement. The fields "first_isomorphic" answers whether the first
statement is isomorphic to the anchor, that is, it can be obtained from it by renaming the
predicates. The fields "second_isomorphic" answers whether the second statement is isomorphic
to the anchor, that is, it can be obtained from it by renaming the predicates.

The "answer" field should contain exactly one of the labels "first", "second" or "draw". The
"first_isomorphic" and "second_isomorphic" fields should be either "YES" or "NO". The value of
"explanation" should be a string explaining your decisions.

ANCHOR STATEMENT
PREMISES
{anchor_premises}
CONCLUSION
{anchor_conclusion}

STATEMENT 1
PREMISES
{premises_1}
CONCLUSION
{conclusion_1}

STATEMENT 2
PREMISES
{premises_2}
CONCLUSION
{conclusion_2}
```

Figure 6: Prompt used to compare instances retrieved by two embedders for the anchor sample.

B Contrastive learning

B.1 Learning objectives

The core idea behind the InfoNCE loss (Oord et al., 2018) is to maximize the similarity of positive pairs by pulling their representations together while pushing apart all other negative samples. This objective is enforced using softmax loss:

$$\mathcal{L}_{i,j} = -\log \frac{\exp(\frac{\text{sim}(\mathbf{z}_i, \mathbf{z}_i^+)}{\tau})}{\exp(\frac{\text{sim}(\mathbf{z}_i, \mathbf{z}_i^+)}{\tau}) + \sum_{k \neq i} \exp(\frac{\text{sim}(\mathbf{z}_i, \mathbf{z}_k)}{\tau})}$$

We implement InfoNCE Loss using MultipleNegativesRankingLoss (Henderson et al., 2017), which treats all other samples in the batch as implicit negatives. This avoids the need for explicit hard negative mining, which is non-trivial in our setting, as structurally distinct logical examples are not straightforwardly identifiable without formal verification. Batch size is critical for InfoNCE quality as it determines the number of in-batch negatives; we use 64, the maximum that fits in the available GPU memory. Full hyperparameters are reported in Table 4.

In our preliminary experiments on FOLIO dataset, wrapping InfoNCE with MatryoshkaLoss (Kusupati et al., 2022) was found to improve the quality of embeddings over plain InfoNCE; we therefore adopt this combination as our primary contrastive training setup. This method computes the loss jointly across multiple embedding subvectors (in our case, of sizes $[1024, 768, 512, 256, 128]$) with equal weights. The nested objective regularizes the model to concentrate discriminative structural information in lower-dimensional subspaces. For inference, the embedder trained with Matryoshka Representation Learning (MRL) is used at dimension 256.

C Infrastructure

We use a single A100 GPU for embedder training and single H100 GPU for inference. Training is done using SentenceTransformers[9] library v 5.1.0 and HuggingFace Transformers [10] v. 4.51.1, while inference is performed with vLLM[11] (Kwon et al., 2023) v 0.12.

[9] https://sbert.net
[10] https://huggingface.co/docs/transformers
[11] vllm.ai

Hyperparameter	Value
Loss (base)	MultipleNegativesRankingLoss
Loss (outer)	MatryoshkaLoss
InfoNCE temperature	0.05
Similarity function	cosine
Matryoshka dims	$[1024, 768, 512, 256, 128]$
Matryoshka weights	$[1, 1, 1, 1, 1]$
Inference dimension	256
Batch size	64
Learning rate	2×10^{-5}
LR scheduler	cosine
Epochs (max)	10
Early stopping patience	2
Warmup ratio	0.1
Weight decay	0.01
Precision	`bfloat16`

Table 4: Hyperparameters for contrastive embedder fine-tuning.

D Datasets

We selected evaluation benchmarks based on the following criteria:

1. NLI format with three-way labeling including an *Unknown* (or *Uncertain*) class, as it is particularly challenging for LLMs and reflects open-world reasoning under incomplete information;
2. deductive logical reasoning in isolation, without reliance on commonsense knowledge or world knowledge;
3. full first-order logic coverage or a well-defined fragment thereof;
4. a fixed training split of sufficient size ($\geq$ 1000 examples) or a reproducible generation pipeline;
5. minimal data conversion overhead.

For all four datasets, analogical examples are generated via DeepSeek-V3 (Liu et al., 2024) rather than through the original dataset pipelines – even in cases, where those pipelines are available. This choice reflects both practical and methodological considerations.

On the practical side: FOLIO examples are human-authored and non-reproducible; the RuleTaker (Clark et al., 2021) generator underlying ProofWriter varies logical structure and lexical content jointly without support for re-verbalizing a fixed proof graph, and its restricted vocabulary ($\sim$100 words) would preclude meaningful semantic variation regardless; the LogicNLI generation pipeline is not publicly available; and for

ProverQA, re-generation would require running the full three-stage pipeline (Prover9 (McCune, 2005) skeleton synthesis, LLM naturalization, and formal verification) per example.

More fundamentally, however, direct manipulation of rule-based generators would yield re-verbalizations constrained to the same templatic vocabulary and syntactic patterns as the originals, which would undermine the intended low surface similarity between pairs. LLM-based generation instead produces naturally varied language across diverse semantic domains while preserving the logical skeleton, which is precisely the property required for effective contrastive training of an embedding model.

Additionally, adopting a single procedure across all datasets ensures that observed differences in retrieval quality reflect properties of the data rather than inconsistencies in the pair construction method.

			Label distribution (%)			
Dataset	**Split**	**Total**	**T**	**F**	**U**	**S**
Original						
FOLIO	train	1,001	38.8	28.9	32.4	–
	val	203	35.5	30.5	34.0	–
LogicNLI	train	16,000	25.0	25.0	25.0	25.0
	val	2,000	25.0	25.0	25.0	25.0
	test	2,000	25.0	25.0	25.0	25.0
ProofWriter	train	585,552	27.1	27.1	45.8	–
	val	85,468	27.1	27.1	45.7	–
	test	174,476	26.6	26.6	46.8	–
ProverQA	train	5,000	42.5	42.1	15.5	–
	val	1,500	31.3	32.7	35.9	–
Final (used in experiments)						
FOLIO	train	900	38.9	28.7	32.4	–
	val	101	37.6	30.7	31.7	–
	test	203	35.5	30.5	34.0	–
LogicNLI	train	16,000	25.0	25.0	25.0	25.0
	val	2,000	25.0	25.0	25.0	25.0
	test	2,000	25.0	25.0	25.0	25.0
ProofWriter	train	8,046	33.3	33.3	33.3	–
	val	982	33.4	33.4	33.2	–
	test	976	33.6	33.2	33.2	–
ProverQA	train	4,000	41.9	42.8	15.3	–
	val	1,000	44.7	39.2	16.1	–
	test	1,500	31.3	32.7	35.9	–

Table 5: Label distribution (%) across original and final (used in experiments) dataset splits. **T** = True/Entailment, **F**= False/Contradiction, **U** = Uncertain/Unknown/Neutral, **S** = Self-Contradiction (LogicNLI only).

D.1 FOLIO (Han et al., 2024)

A crowdsourced FOL-NLI benchmark in which premises and conclusions are written and annotated by trained logicians in natural, semantically rich English, covering the full range of FOL connectives and quantifiers, including universal and existential quantification, negation, disjunction, and nested implications, with reasoning chains of varying depth. In contrast to the other datasets in this study, which are synthetically generated from templatic grammars with restricted vocabularies, FOLIO is the only human-authored benchmark in our evaluation, making it both the most linguistically diverse and the most constrained in size.

The HuggingFace version contains $1,204$ examples in total: a training split of $1,001$ examples and a validation split of 203 examples. As the official test split labels are withheld, we use the validation split as our test set, consistent with prior work. From the training split, we use 900 examples as the few-shot retrieval pool and for embedder fine-tuning, holding out the remaining 101 as the embedder validation set, explicitly replicating the low-resource setting that motivates the present study.

D.2 ProofWriter (Tafjord et al., 2021)

A formally verified deductive reasoning benchmark based on the RuleTaker framework, extended with an *Unknown* label under the open-world assumption and explicit proof chains. Premises are synthetically generated from a restricted Datalog (Gottlob et al., 1989) fragment of FOL (universal quantification, conjunctive antecedents, and iterative modus ponens) with a templatic vocabulary of approximately 100 words; disjunction and existential quantifiers are absent. The generation pipeline is fully open and allows arbitrary scaling. The aggregated HuggingFace version combines all configuration types (`AttNeg`, `AttNoneg`, `AttNonegNatLang`, `RelNeg`, `RelNoneg`) across proof depths $D0$–$D5$, where each theory generates three to four questions with different labels, yielding $585,552$, $85,468$, and $174,476$ (theory, question, answer) triples in the train, validation, and test splits respectively.

We apply a multi-stage preprocessing pipeline to reduce these to tractable and balanced subsets. First, examples with anomalous proof depths (QDep > 5) are removed. Each split is then balanced by jointly stratifying over label class (*True*/*False*/*Unknown*) and proof depth QDEP,

sampling uniformly across all (label × QDEP) cells bounded by the minimum cell count; a finer stratified subsample is subsequently applied using a composite key (`config_type` × label × QDEP), targeting sizes of $8,000/1,000/1,000$ for train/validation/test respectively. Finally, to prevent cross-split contamination, strict theory-level disjointness is enforced across splits: any theory present in a higher-priority split is removed from all lower-priority ones. Due to the granularity of stratification, the resulting sizes differ slightly from the targets; after deduplication the final counts are $8,046/982/976$ for train/validation/test respectively, with the training split additionally serving as the few-shot retrieval pool.

D.3 ProverQA (Qi et al., 2025)

A recently released benchmark generated via a three-stage pipeline: logical skeleton synthesis via Prover9, controlled naturalization via LLM, and formal answer verification. Compared to ProofWriter, ProverQA features substantially more natural language while retaining formal correctness guarantees. The test split ($1,500$ examples) is stratified by difficulty level – easy, medium, and hard (500 each), thus ensuring controlled evaluation across reasoning complexity. The training split ($5,000$ examples) is provided in SFT format; we convert it to NLI format by parsing the instruction field and mapping answer options to True/False/Uncertain labels.

For embedder fine-tuning and few-shot pool construction, we sample $4,000$ and $1,000$ examples for training and evaluation, respectively, from the original training set via group-based splitting on premise identity to prevent cross-split contamination.

D.4 LogicNLI (Tian et al., 2021)

A synthetically generated FOL-NLI benchmark with four labels: *entailment*, *contradiction*, *neutral*, and *self-contradiction*, where the last class covers cases in which both a statement and its negation are derivable from the premise set, which is a structurally distinct failure mode from standard contradiction. Full FOL coverage is provided, including all seven basic connectives and quantifiers, with reasoning chains up to five steps. The language is semi-templatic and grammatically limited in places, which is a known limitation of the dataset, partially compensated by the breadth of logical constructions covered. The generation pipeline is not publicly released; only the data is available.

The dataset provides fixed train/validation/test splits of $16,000/2,000/2,000$ examples, which we use without any modifications, as opposed to the other datasets in this study.

Hyperparameter	Value
Training method	SFT (completion-only loss)
Batch size	4
Gradient accumulation	4
Effective batch size	16
Learning rate	2×10^{-5}
LR scheduler	cosine
Epochs (max)	5
Early stopping patience	2
Warmup ratio	0.05
Weight decay	0.01
Max sequence length	4096
Precision	`bfloat16`
Thinking token (reasoning)	`<think>\n\n</think>`

Table 6: Hyperparameters for supervised fine-tuning of inference models.

E Additonal ablations

We perform two additional ablation experiments:

1. As FOL correctness is invariant under the order of premises, it is natural to ask whether shuffling the order of premises in the positive samples further improves the quality of embeddings. As shown on Fig. 7 (FT and vs FT+perm.), this modification does not further outperform the default tuned retriever but still lies above the pretrained embedder.

2. Modern embedders like Qwen3-Embedding were trained to follow instructions, so desirable properties of embeddings might be induced by prompting, not by tuning. Comparing the non-prompted (PT) and prompted (PT+prompt) versions of embedder on Fig. 7, we observe that formulating the similarity objective via task instruction provides benefits over the baseline for 4B models but is unable to compete with embedder tuning.

Method	4B-INSTRUCT				4B-THINKING				8B			
	F	LN	PW	PQ	F	LN	PW	PQ	F	LN	PW	PQ
Zero	62.0	40.0	64.1	61.3	68.0	36.4	65.0	67.9	77.8	48.9	86.3	80.9
Random	**70.4**	38.1	**79.3**	69.3	75.9	36.3	76.8	75.9	**83.7**	49.0	**89.5**	82.7
PT	62.0	38.2	76.3	68.7	74.8	34.7	80.0	76.5	78.3	49.7	87.0	82.3
FT	68.3	**39.6**	77.7	**70.9**	**77.6**	**36.7**	80.5	**78.1**	80.7	**49.9**	88.1	**83.6**
Full FT	77.3	77.3	92.3	87.8	81.3	77.3	94.0	89.2	66.5	69.7	92.9	88.8

Table 7: Accuracy (%) across datasets and Qwen3 models (F=FOLIO, LN=LogicNLI, PW=ProofWriter, PQ=ProverQA) with `e5-base-v2` as the backbone embedder (PT = pretrained, FT = contrastively finetuned). **Bold**: best among few-shot methods (rows 2–4) per column. *Italics*: FT embedder does not exceed PT embedder. Underline: full fine-tuning underperforms best few-shot configuration.

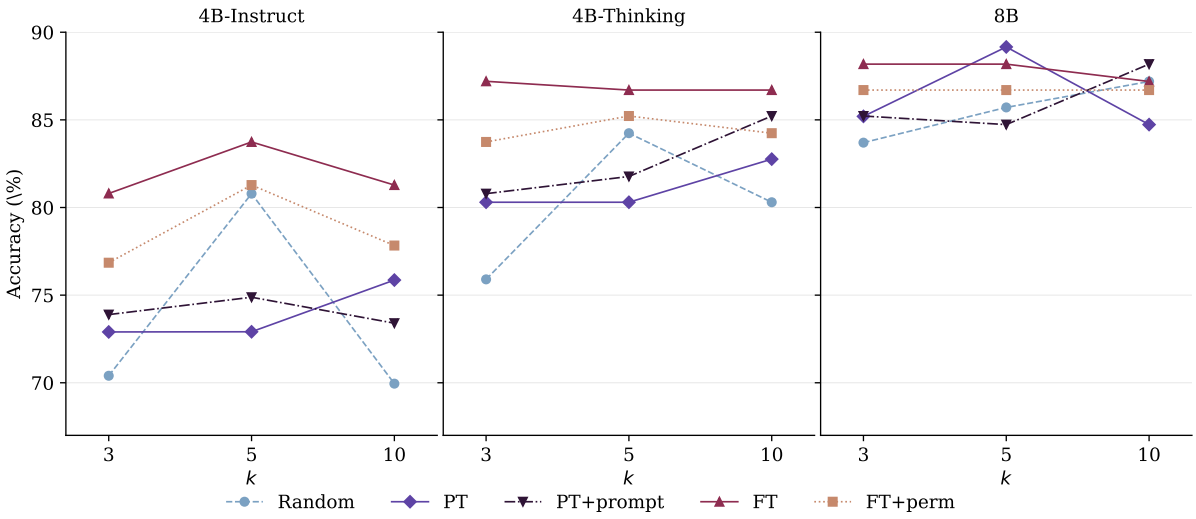

Figure 7: FOLIO accuracy (%) for varying k-shot settings across three Qwen3 model sizes and five retrieval strategies: random sampling, pre-trained embedder (PT), instruction-prompted pre-trained embedder (PT+prompt), finetuned embedder (FT), and embedder finetuned with permuted premises (FT+perm).

Method	4B-INSTRUCT			4B-THINKING			8B		
	k=3	k=5	k=10	k=3	k=5	k=10	k=3	k=5	k=10
Random	70.4	80.8	70.0	75.9	84.2	80.3	83.7	85.7	87.2
PT	72.9	72.9	75.9	80.3	80.3	82.8	85.2	**89.2**	84.7
FT	**80.8**	**83.7**	**81.3**	**87.2**	**86.7**	**86.7**	**88.2**	88.2	87.2
FT+perm	76.9	81.3	77.8	83.7	85.2	84.2	86.7	86.7	86.7
PT+prompt	73.9	74.9	73.4	80.8	81.8	85.2	85.2	84.7	**88.2**

Table 8: FOLIO accuracy (%) across retrieval strategies and k-shot settings for three Qwen3 model sizes. The figure 7 provides the corresponding visual comparison. **Bold**: best per model$\times k$-shot combination.