Does My Embedding Reflect That $A = B$? Evaluating Mathematical Equivalence in Embedding Models

Anonymous ACL submission

Abstract

Because mathematics is highly abstract, a single statement can take very different forms depending on what subfield it is framed in. There are many examples where mathematics progressed only after researchers discovered that a result had already been proven in a different field, using very different language. At the same time, the growth of new resources related to formalization has increased the need for tools that enable efficient and reliable navigation between mathematical 'languages' (e.g., from Lean to natural language). In this paper, we investigate whether current embedding models can capture mathematical equivalence. To do this, we introduce the *Mathematically Equivalent but Lexically Different Pairs (MELD) Dataset*, a collection of paired statements that are expressed in very different language but are mathematically equivalent. We show that the current state-of-the-art embedding models struggle with this task. Motivated by this, we propose a contrastive approach to learning embeddings of mathematical text that focuses on aligning different representations. We specifically train models to align informal statements with different formalizations. Our experiments demonstrate that this leads to improvements not only on informal-formal retrieval tasks but also on MELD, which only contains natural language statements.

1 Introduction

Language models have become increasingly capable of research-level mathematics with several documented cases of LLMs like GPT (Tao, 2024; Gowers, 2026; OpenAI, 2026), Gemini (Feng et al., 2026), and Claude (Knuth, 2026) helping mathematicians make progress on open problems. One of the advantages of these systems is their ability to draw on the vast mathematics literature (through both parametric knowledge and internet search) at a scale that is inaccessible to humans. This includes making non-trivial connections that extend beyond surface-level lexical overlap.

At the same time, the ability to effectively navigate large collections of mathematical text is becoming essential not only because of the ever-growing number of mathematics papers available in online databases like the arXiv, but also because of new formalization-driven resources like `mathlib` (The mathlib Community, 2020). Although state-of-the-art LLMs may already have the capability to solve these tasks, they are too expensive computationally to scale to many applications. On the other hand, text embedding models are explicitly designed to capture semantic relationships within a corpus of text efficiently. In this paper, we ask whether current state-of-the-art embedding models can capture mathematical equivalence.

To test this, we introduce a new dataset, *Mathematically Equivalent Lexically Different Pairs (MELD)*, a collection of 270 pairs of statements that use different terminology but are mathematically equivalent. One example is the definition of a product of two sets in terms of set theory versus the same notion defined via category theory. We then show that state-of-the-art embedding models tend not to embed these mathematically equivalent pairs near one another, instead grouping statements based on the mathematical subfield they are framed in.

An embedding model that could group statements based on mathematical equivalence rather than lexical similarity would be a powerful tool for mathematical discovery. Motivated by this, we propose a contrastive approach to training embedding models for mathematics that uses formal and informal versions of a statement as different "views" of the underlying mathematics. To realize this, we generate an augmented version of the `mathlib_informal_v4.16.0` dataset (FrenzyMath, 2025a), which contains Lean statements from the `mathlib` library along with

informal descriptions. We supplement the original dataset with rephrasings of the informal descriptions. The two models we train under this setup, `MathLeap-Qwen-8B` and `MathLeap-Octen-8B`, achieve strong results across a range of mathematics retrieval tasks[1]. Supporting our hypothesis that our approach to training encourages a model to learn representations that reflect the mathematical content of a statement, we show that both models achieve the highest scores on MELD from among the models that we evaluated. This is particularly notable since MELD only contains natural language content.

In summary, our contributions include:

1. We introduce *Mathematically Equivalent but Lexically Different Pairs (MELD)*, a dataset with 270 pairs of mathematical statements that are lexically dissimilar but mathematically equivalent.

2. Observing that most embedding models prioritize language similarity over mathematical equivalence, we propose a contrastive approach to developing embedding models for mathematics, using informal and formal representations of a statement as different views of the underlying mathematical content.

3. We show that this leads to significant improvements on MELD and a range of other mathematical retrieval tasks.

2 Do Embedding Models Capture Mathematical Equivalence?

One of the most attractive features of mathematics (and also one of the features that can make mathematics so hard) is that a single statement can often be described precisely in two radically different ways. These "one thing, two perspectives" situations have historically been the foundation of many fruitful areas of mathematics (e.g., algebraic geometry, algebraic topology, and geometric group theory).

Motivated by this, we introduce *Mathematically Equivalent but Lexically Different Pairs (MELD)*, a collection of 270 statement pairs written in natural language and LaTeX where paired statements are equivalent but cannot be easily matched by surface-level lexical similarity. One example is the following:

> **Identity map (Set theory)** For a set A, $\mathrm{id}_A : A \to A$ defined by $\mathrm{id}_A(a) = a$.
>
> **Identity map (Category theory)** For each object A, a distinguished arrow $\mathrm{id}_A : A \to A$ such that $f \circ \mathrm{id}_A = f$ and $\mathrm{id}_A \circ g = g$ for any objects B and C with $f : A \to B$ and $g : C \to A$.

The 270 pairs are grouped into nine different pairs of domains that offer distinctive descriptions of a common set of ideas (e.g., representation theory and Fourier analysis). These are summarized in Table 3.

We generated MELD by iterating through each of the nine pairs of complementary domains, describing the connection between the two fields, and prompting Claude Opus 4.7 to generate 30 pairs of mathematically equivalent but lexically distinct statements. We then manually reviewed these, making modifications to increase dissimilarity while preserving mathematical equivalence. We then evaluated all statements using GPT-5.5 (medium) to check (i) whether both statements were valid, (ii) whether they were equivalent, and (iii) whether they could be made to sound less similar.

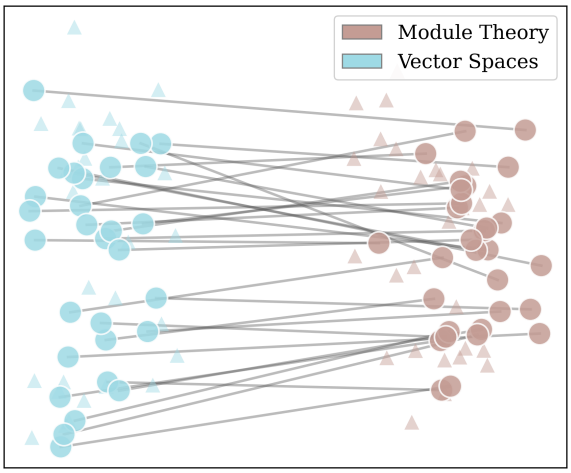

Figure 1: A UMAP visualization of embedded statements from the "vector space theory vs module theory" subset of MELD. Blue points correspond to statements framed in terms of vector space terminology, brown points correspond to statements framed in terms of module theory. Small triangles are distractor statements. MELD pairs are large dots connected by lines. Embedded statements cluster by subfield rather than mathematical equivalence.

[1]Models and datasets can be found at https://anonymous-hf.up.railway.app/a/xczmsjlqejkf/, https://anonymous-hf.up.railway.app/a/pv25ongyl2qb/ and https://anonymous-hf.up.railway.app/a/9n9cngyu38hk/ . Code can be found at https://anonymous.4open.science/r/submission-repo/.

Table 1: Retrieval recall of mathematically equivalent statements from MELD.

Model	Recall@1	Recall@3	Recall@5	Recall@10	Recall@20
Qwen3-Embedding-4B	13.7	34.3	45.0	58.5	70.2
Qwen3-Embedding-8B	17.0	38.0	47.8	63.0	78.1
harrier-oss-v1-27b	19.4	37.0	48.3	63.9	78.5
KaLM-Embedding-Gemma3-12B	10.4	27.4	35.2	48.9	64.6
llama-embed-nemotron-8b	2.8	7.2	11.1	19.3	31.7
Octen-Embedding-8B	25.0	50.6	62.8	77.4	88.9
MathLeap-Qwen-8B (ours)	27.2	52.2	63.1	**78.0**	**89.8**
MathLeap-Octen-8B (ours)	**28.9**	**55.0**	**63.7**	77.2	**89.8**

While using LLM-generated datasets carries risks, all content belongs to a basic graduate curriculum in mathematics, and hence should be well within the capabilities of current LLMs. This also means that embedding model failures on MELD is less likely to be attributable to the subject matter being esoteric and more likely to point toward limitations in the way that these models represent mathematical text.

To understand the extent to which embeddings of mathematical text are organized by mathematical equivalence rather than just lexical similarity, we embed all 540 statements using some of the best current embedding models and look at whether equivalent pairs are grouped near each other. This includes `Qwen3-Embedding-4B` (Zhang et al., 2025), `Qwen3-Embedding-8B` (Zhang et al., 2025), `harrier-oss-v1-27b` (Microsoft, 2026), `KaLM-Embedding-Gemma3-12B` (Hu et al., 2025; Zhao et al., 2025), `llama-embed-nemotron-8b` (Babakhin et al., 2025), and `Octen-Embedding-8B` (Team, 2025).

We find that statements are mostly clustered by the field they are expressed in. For instance, in Table 1 the model that achieves the best results, Octen-Embedding-8B, has a recall@1 of only 0.241. To get a qualitative sense of why this is, in Figure 1 we use UMAP (McInnes et al., 2018) to reduce the dimensionality of the embedding and then visualize statements from 30 pairs where one element is written in the language of vector spaces (blue) and the other is written in the language of module theory (brown). Paired statements are connected by gray lines. The model appears to strongly clusters statements by the field rather than mathematical equivalence.

3 A Contrastive Approach to Embedding Mathematical Text

The experiments in Section 2 suggest that current embedding models capture the lexical similarity of mathematical statements rather than their mathematical equivalence. Motivated by this, we consider training strategies that steer models toward embeddings that cluster statements by mathematical equivalence rather than surface-level similarities in language. While training on MELD has the potential to do this, it is small and there is not an easy way to scale it since it depends on state-of-the-art LLMs. Fortunately, the growth of formalization has begun to create resources that address this need.

`mathlib` is essential for formal proof development in Lean because it provides formalizations of central definitions and theorems from modern mathematics. `mathlib` was used to generate the `mathlib_informal_v4.16.0` dataset (FrenzyMath, 2025a). Each entry in this dataset corresponds to a `mathlib` statement, which comes in three different forms: the Lean signature, the Lean type, and a natural language translation. We treat these as different "views" on a single underlying mathematical statement. Our assumption is that by steering a model toward alignment of formal and informal versions of statements, we will produce embeddings that better capture mathematical equivalence.

3.1 Training Our MathLeap Models

In addition to the natural language, Lean signature, and Lean type views inherited from `mathlib_informal_v4.16.0`, we also generate natural language rephrasings for some instances in the training set. We do this by first using `Qwen3.5-9B` to decompose natural language theorems into explicit hypotheses and conclusions. We then provide these to `gemma-4-E4B-it` to generate rephrasings. When doing this, `gemma-4-E4B-it` is prompted to change superficial aspects of the theorem statement like variable names. More details, including the prompts that were used, can be found in Section C.1. Our final training set contains 118,334 instances (with three to four views each)

while our test set contains 15,287 instances. Approximately 85% of the training set had a natural language rephrasing attached to it.

Since our initial experiments showed that `Qwen3-Embedding-8B` (Zhang et al., 2025) and `Octen-Embedding-8B` (Team, 2025) are effective at capturing mathematical equivalence relative to comparable embedding models, we chose to use these as base models for finetuning. We trained using Cached Multiple Negatives Ranking Loss from the Sentence-BERT library (Reimers and Gurevych, 2019), a contrastive loss function that uses in-batch negatives while avoiding common issues that arise when trying to scale contrastive methods to GPU training. We used the default scale value of 20.0. We used AdamW, 700 warm-up steps, a constant scheduler, a batch size of 16, and a max sequence length of 128. Models were trained for approximately two epochs ($\sim$14,790 steps). Training was performed on a single H200 and took approximately 8 hours. We call the resulting models `MathLeap-Qwen-8B` and `MathLeap-Octen-8B` respectively.

3.2 Evaluation

We evaluate `MathLeap-Qwen-8B` and `MathLeap-Octen-8B` on a range of mathematical retrieval tasks. Overall, we find that our contrastive approach to training a mathematics embedding model significantly improves performance on tasks where retrieving mathematically equivalent statements is critical.

MELD Finetuning `Qwen3-Embedding-8B` results in recall@1 increasing from 17.0 to 27.2 and recall@5 increasing from 47.8 to 63.1 (Table 1). At the same time, the average rank of the mathematically equivalent pair among all neighbors decreases from 16.1 to 9.7 (Table 5). Both `MathLeap-Qwen-8B` and `MathLeap-Octen-8B` independently score higher on MELD than any of the other models we evaluated on each of the metrics we considered. This is particularly notable since natural-language-only retrieval is a small part of the training curriculum of `MathLeap-Qwen-8B`. We see this as supporting our hypothesis that our contrastive style of training does not just help with the specific task of aligning representations of formal and informal mathematical statements, it also encourages representations that capture mathematical equivalence.

Blueprints Lean blueprints are dependency graphs used to track progress on formalization projects. Nodes in these dependency graphs typically contain both a Lean statement and a natural language description. We used these pairs across 68 graphs to create a dataset of 3,729 Lean and natural language pairs. We measured recall on retrieval of the corresponding Lean statement given a natural language query. Across all models that we evaluated, `MathLeap-Qwen-8B` or `MathLeap-Octen-8B` performed best across the evaluated recall metrics (Table 5). This is not as surprising as our results on MELD, since the blueprints task aligns closely with the training task for both MathLeap models.

MIRB We evaluate `MathLeap-Qwen-8B` and `MathLeap-Octen-8B` on three tasks from the MIRB benchmark: MathOverflow duplicate question retrieval, `mathlib` retrieval, and Lean premise retrieval (originally from (Yang et al., 2023)). We measure performance using nDCG@10, following (Ju and Dong). `MathLeap-Octen-8B` performs best on `mathlib` retrieval, which is well-aligned with its training set. On the other hand, we see reduced performance on the other two tasks. We analyze these trade-offs in the Limitations section below.

4 Conclusion

The ability of an embedding model to capture mathematical equivalence rather than just surface-level textual similarity is important to mathematical discovery. In this paper, we ask whether today's state-of-the-art embedding models are capable of retrieving mathematically equivalent statements from among lexically similar distractors. Using a new dataset, Mathematically Equivalent but Lexically Different Pairs (MELD), we find that current models struggle to do this. This leads us to propose a contrastive approach to mathematics-specific embedding model training using informal and formal statements as different views of the underlying mathematics. Using this approach, we train `MathLeap-Qwen-8B` and `MathLeap-Octen-8B`. These models display substantial improvement across many mathematics retrieval tasks and suggest a path toward training even stronger mathematics specific embedding models in the future.

Limitations

There are a number of limitations to this work that should be highlighted. Most importantly,

we note that on some tasks such as MathOverflow duplication retrieval and Lean premise retrieval, the performance of `MathLeap-Qwen-8B` and `MathLeap-Octen-8B` decreases relative to their base models. We suspect this is due to our models becoming over-specialized during finetuning. For example, MathOverflow duplication retrieval may require a broader natural-language understanding outside of rigorous theorem statements, an ability that may degrade in our models during finetuning. On the other hand, Lean premise retrieval involves using a proof-state as a query and `mathlib` declarations as the corpus. Our training set does not include any proof-states so it is not surprising that performance degrades. Overall, this points to the need for a more diverse training set if our goal is a generalist math embedding model.

We also note that we depend on synthetic data for a number of our datasets. MELD depends on frontier LLMs for identification and generation of basic theorems and definitions, as well as translation of these to the language of a different mathematical subfield. The training set of `MathLeap-Qwen-8B` and `MathLeap-Octen-8B` uses rephrasing generated by local models. In both cases, we have attempted to remove any erroneous instances, but we cannot guarantee that all errors have been eliminated.

References

Luke Alexander, Eric Leonen, Sophie Szeto, Artemii Remizov, Ignacio Tejeda, Jarod Alper, Giovanni Inchiostro, and Vasily Ilin. 2026. Semantic search over 9 million mathematical theorems. *arXiv preprint arXiv:2602.05216*.

Yauhen Babakhin, Radek Osmulski, Ronay Ak, Gabriel Moreira, Mengyao Xu, Benedikt Schifferer, Bo Liu, and Even Oldridge. 2025. Llama-embed-nemotron-8b: A universal text embedding model for multilingual and cross-lingual tasks. *Preprint*, arXiv:2511.07025.

Tony Feng, Trieu Trinh, Garrett Bingham, Jiwon Kang, Shengtong Zhang, Sang-hyun Kim, Kevin Barreto, Carl Schildkraut, Junehyuk Jung, Jaehyeon Seo, and 1 others. 2026. Semi-autonomous mathematics discovery with Gemini: A case study on the Erdős problems. *arXiv preprint arXiv:2601.22401*.

FrenzyMath. 2025a. mathlib_informal_v4.16.0.

FrenzyMath. 2025b. mathlib_informal_v4.19.0.

Timothy Gowers. 2026. A recent experience with ChatGPT 5.5 Pro. Gowers's Weblog. Discussion on LLMs solving research-level problems.

Xinshuo Hu, Zifei Shan, Xinping Zhao, Zetian Sun, Zhenyu Liu, Dongfang Li, Shaolin Ye, Xinyuan Wei, Qian Chen, Baotian Hu, Haofen Wang, Jun Yu, and Min Zhang. 2025. Kalm-embedding: Superior training data brings a stronger embedding model. *Preprint*, arXiv:2501.01028.

Haocheng Ju and Bin Dong. MIRB: Mathematical information retrieval benchmark. In *2nd AI for Math Workshop@ ICML 2025*.

Donald E. Knuth. 2026. Recent news. Donald Knuth's Home Page. Accessed: 2026-05-14.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Microsoft. 2026. Harrier-oss-v1-27b. `https://huggingface.co/microsoft/harrier-oss-v1-27b`. Hugging Face model card.

OpenAI. 2026. Planar point sets with many unit distances. *OpenAI Technical Report*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Terence Tao. 2024. AI is ready for primetime in math and theoretical physics. OpenAI Academy Blog / Terence Tao's Home Page. Accessed: 2026-05-14.

Octen Team. 2025. Octen series: Optimizing embedding models to 1 on rteb leaderboard.

The mathlib Community. 2020. The Lean Mathematical Library. In *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs*, CPP 2020, New Orleans, LA, USA. ACM.

Sean Welleck, Jiacheng Liu, Ronan Le Bras, Hannaneh Hajishirzi, Yejin Choi, and Kyunghyun Cho. 2021. Naturalproofs: Mathematical theorem proving in natural language. *arXiv preprint arXiv:2104.01112*.

Kaiyu Yang, Aidan Swope, Alex Gu, Rahul Chalamala, Peiyang Song, Shixing Yu, Saad Godil, Ryan J Prenger, and Animashree Anandkumar. 2023. Leandojo: Theorem proving with retrieval-augmented language models. *Advances in Neural Information Processing Systems*, 36:21573–21612.

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

Xinping Zhao, Xinshuo Hu, Zifei Shan, Shouzheng Huang, Yao Zhou, Xin Zhang, Zetian Sun, Zhenyu Liu, Dongfang Li, Xinyuan Wei, Youcheng Pan, Yang Xiang, Meishan Zhang, Haofen Wang, Jun Yu, Baotian Hu, and Min Zhang. 2025. Kalm-embedding-v2: Superior training techniques and data inspire a versatile embedding model. *Preprint*, arXiv:2506.20923.

A Related work

The value of being able to efficiently leverage the mathematics literature has meant that retrieval and search have been an active area of research in the AI-for-math community. For example, (Welleck et al., 2021) introduced a dataset of mathematical theorems, treating theorem search as a reference retrieval task. This showed that text embedding models can be effective when retrieval only depends on capturing surface-level similarities between mathematical statements but may be less effective when deeper mathematical connections are required. More recently, (Alexander et al., 2026) used embeddings from Qwen3-Embedding-8B to perform retrieval over 9 million theorems. Our work aims to improve the capabilities of text embedding models themselves, focusing specifically on capturing mathematical equivalence.

More recently (Ju and Dong), proposed a full text embedding benchmark for mathematical retrieval, Mathematics Information Retrieval Benchmark (MIRB). This benchmark includes tasks that mix less rigorous language (e.g., search for Math Stack Exchange duplicates) to formal Lean premise retrieval. Our work differs from MIRB, as MELD is designed to specifically target mathematical equivalence with the ultimate goal of designing better embedding models.

Table 2: MMR and mean pair rank for embedding models on MELD.

Model	MMR	Mean rank
Qwen3-Embedding-4B	0.28	21.4
Qwen3-Embedding-8B	0.32	16.1
harrier-oss-v1-27b	0.33	15.2
KaLM-Embedding-Gemma3-12B-2511	0.23	25.6
llama-embed-nemotron-8b	0.08	94.7
Octen-Embedding-8B	0.42	10.3
F2LLM-v2-14B	0.37	10.17
MathLeap-Qwen-8B (ours)	0.43	9.7
MathLeap-Octen-8B (ours)	**0.45**	**9.3**

Table 3: Subfield pairs used to generate MELD

Mathematical area 1	Mathematical area 2
Vector spaces	Module theory
Measure theory	Probability theory
Set theory	Category theory
Geometry	Commutative algebra
Algebraic topology	Group theory
Graph theory	Linear algebra
Combinatorics	Generating functions
Representation theory	Fourier analysis
Symmetric functions	Tableaux

Table 4: Retrieval scores on the blueprints dataset. Retrieval was performed by giving the informal description and asking the model to find the corresponding Lean statement.

Model	Recall@1	Recall@5	Recall@10
harrier-oss-v1-27B	55.00	85.82	92.07
Qwen3-Embedding-4B	56.94	88.76	94.11
Qwen3-Embedding-8B	59.52	90.13	94.11
KaLM-Embedding-Gemma3-12B-2511	54.04	86.02	91.96
llama-embed-nemotron-8b	23.80	44.11	52.70
Octen-Embedding-8B	64.32	92.60	96.37
MathLeap-Qwen-8B (ours)	**64.73**	**93.10**	96.44
MathLeap-Octen-8B (ours)	64.69	92.80	**96.92**

Table 5: nDCG@10 scores for three MIRB tasks.

Model	MO Dup.	`mathlib` Retrieval	Lean Premise
harrier-oss-v1-0.6B	0.762	0.345	0.087
Qwen3-Embedding-4B	0.855	0.531	0.132
Qwen3-Embedding-8B	**0.859**	0.559	0.136
KaLM-Embedding-Gemma3-12B-2511	0.778	0.504	0.133
llama-embed-nemotron-8b	0.730	0.267	0.050
Octen-Embedding-8B	0.855	0.653	**0.143**
MathLeap-Qwen-8B (ours)	0.821	0.653	0.103
MathLeap-Octen-8B (ours)	0.816	**0.667**	0.103

B FrenzyMath Retrieval Benchmark

We report the full paired-retrieval benchmark used to evaluate alignment between informal mathematical language and Lean representations. The benchmark is derived from `FrenzyMath/mathlib_informal_v4.19.0` (FrenzyMath, 2025b), which provides aligned informal descriptions, Lean types, and Lean signatures for mathlib declarations. To focus on mathematics-focused declarations, we retained examples whose `module_name` field has a second component in the following set: Algebra, AlgebraicGeometry, AlgebraicTopology, Analysis, CategoryTheory, Combinatorics, Computability, Condensed, Control, Data, Dynamics, FieldTheory, Geometry, GroupTheory, InformationTheory, LinearAlgebra, Logic, MeasureTheory, ModelTheory, NumberTheory, Order, Probability, RepresentationTheory, RingTheory, SetTheory, and Topology. This filter reduces the upstream training split from 218,071 declarations to 211,237 declarations.

We preserve the upstream dataset order after filtering and construct the benchmark splits with the Hugging Face `train_test_split` routine. First, we split the filtered corpus with `test_size=0.2` and `seed=42`, yielding the training split and a held-out remainder. We then split that held-out remainder again with `test_size=0.5` and `seed=42`, yielding validation and test splits of equal size. The final split sizes are 168,989 training examples, 21,124 validation examples, and 21,124 test examples.

Each run is an exact paired-retrieval task. Queries are drawn from the held-out test split, and the retrieval corpus is the union of the train, validation, and test splits. A retrieved item is counted as correct only if it matches the row-aligned target for the query. Tasks involving Lean signatures evaluate 20,951 test queries after examples with empty signatures are excluded, while tasks involving Lean types evaluate all 21,124 test queries. All embeddings are L2-normalized, and retrieval is performed by exact nearest-neighbor search in the shared embedding space.

Tables 6-11 report six directed retrieval tasks: informal description → Lean type, Lean type → informal description, informal description → Lean signature, Lean signature → informal description, Lean type → Lean signature, and Lean signature → Lean type. For instruction-aware models, the task instruction is applied only to the query side, while the corpus side is embedded as the fixed candidate bank. We report Recall@1 (R@1), Recall@5 (R@5), Recall@10 (R@10), and mean reciprocal rank (MRR) for the base embedding models used in the main experiments and our fine-tuned variants, `MathLeap-Qwen-8B` and `MathLeap-Octen-8B`.

Table 6: Informal description → Lean type.

Model	R@1	R@5	R@10	MRR
MathLeap-Qwen-8B	**69.34%**	**88.35%**	**91.72%**	**77.77%**
MathLeap-Octen-8B	**70.18%**	**88.27%**	**91.64%**	**78.26%**
Octen-Embedding-8B	64.06%	87.91%	91.43%	74.47%
Qwen3-Embedding-8B	61.89%	86.70%	90.71%	72.65%
Qwen3-Embedding-4B	57.01%	84.33%	89.08%	68.79%
F2LLM-v2-14B	67.97%	90.42%	93.36%	77.78%
Harrier-OSS-v1-27B	62.26%	87.90%	91.67%	73.37%
Llama-Embed-Nemotron-8B	27.87%	52.79%	61.72%	38.42%
KaLM-Gemma3-12B	56.32%	82.25%	87.49%	67.54%

Table 7: Lean type → informal description.

Model	R@1	R@5	R@10	MRR
MathLeap-Qwen-8B	**65.22%**	**85.77%**	**89.29%**	**74.26%**
MathLeap-Octen-8B	**66.89%**	**86.54%**	**89.92%**	**75.57%**
Octen-Embedding-8B	58.17%	85.56%	90.07%	69.90%
Qwen3-Embedding-8B	44.97%	74.90%	82.61%	57.69%
Qwen3-Embedding-4B	45.33%	75.00%	82.40%	57.98%
F2LLM-v2-14B	56.78%	85.02%	90.09%	68.93%
Harrier-OSS-v1-27B	33.03%	61.87%	71.53%	45.33%
Llama-Embed-Nemotron-8B	1.17%	3.56%	5.78%	2.30%
KaLM-Gemma3-12B	42.31%	71.30%	79.43%	54.69%

Table 8: Informal description → Lean signature.

Model	R@1	R@5	R@10	MRR
MathLeap-Qwen-8B	**67.40%**	**88.75%**	**91.96%**	**76.64%**
MathLeap-Octen-8B	**67.24%**	**87.82%**	**91.51%**	**76.28%**
Octen-Embedding-8B	56.08%	79.47%	84.27%	66.28%
Qwen3-Embedding-8B	54.64%	77.95%	82.90%	64.79%
Qwen3-Embedding-4B	49.14%	75.22%	81.18%	60.37%
F2LLM-v2-14B	57.73%	81.16%	85.94%	67.98%
Harrier-OSS-v1-27B	54.16%	78.79%	84.04%	64.76%
Llama-Embed-Nemotron-8B	21.89%	39.42%	46.20%	29.47%
KaLM-Gemma3-12B	47.56%	72.83%	79.07%	58.31%

Table 9: Lean signature → informal description.

Model	R@1	R@5	R@10	MRR
MathLeap-Qwen-8B	**71.28%**	**91.98%**	**94.45%**	**80.33%**
MathLeap-Octen-8B	**72.77%**	**92.33%**	**94.77%**	**81.37%**
Octen-Embedding-8B	57.44%	82.94%	87.94%	68.48%
Qwen3-Embedding-8B	42.39%	70.67%	78.16%	54.50%
Qwen3-Embedding-4B	47.10%	73.93%	80.39%	58.65%
F2LLM-v2-14B	57.51%	84.36%	89.47%	69.10%
Harrier-OSS-v1-27B	38.47%	66.16%	74.51%	50.29%
Llama-Embed-Nemotron-8B	2.62%	6.73%	9.66%	4.43%
KaLM-Gemma3-12B	43.36%	70.78%	78.09%	55.12%

Table 10: Lean type → Lean signature.

Model	R@1	R@5	R@10	MRR
MathLeap-Qwen-8B	**65.01%**	**83.83%**	**87.72%**	**73.24%**
MathLeap-Octen-8B	**65.26%**	**84.29%**	**88.22%**	**73.61%**
Octen-Embedding-8B	60.40%	81.99%	86.60%	69.81%
Qwen3-Embedding-8B	58.00%	80.41%	85.08%	67.75%
Qwen3-Embedding-4B	51.99%	77.28%	83.13%	62.90%
F2LLM-v2-14B	57.96%	81.90%	86.82%	68.33%
Harrier-OSS-v1-27B	48.36%	72.91%	79.04%	58.97%
Llama-Embed-Nemotron-8B	8.81%	15.90%	19.02%	11.84%
KaLM-Gemma3-12B	45.43%	68.88%	75.45%	55.48%

Table 11: Lean signature → Lean type.

Model	R@1	R@5	R@10	MRR
MathLeap-Qwen-8B	**72.46%**	**89.01%**	**92.06%**	**79.82%**
MathLeap-Octen-8B	**73.34%**	**89.68%**	**92.59%**	**80.56%**
Octen-Embedding-8B	70.04%	89.27%	92.31%	78.45%
Qwen3-Embedding-8B	69.08%	89.10%	92.50%	77.88%
Qwen3-Embedding-4B	64.41%	86.24%	90.19%	73.98%
F2LLM-v2-14B	70.43%	90.41%	93.63%	79.22%
Harrier-OSS-v1-27B	61.74%	84.45%	88.69%	71.52%
Llama-Embed-Nemotron-8B	16.57%	28.67%	33.59%	21.79%
KaLM-Gemma3-12B	67.16%	87.33%	90.83%	76.03%

C Training data generation

C.1 Prompts for theorem decomposition, rephrasing, and hard negative generation

In this section we provide the prompts that we used for generate the natural language rephrasings and hard negatives. The first step in this process was to down select to theorems in `FrenzyMath/mathlib_informal_v4.16.0`. We then had `Qwen/Qwen3.5-9B` decompose each of these into assumptions and conclusions using the following prompt:

```
You are a mathematical statement parser.

Task:
Given a single research-level
mathematical statement
(theorem, lemma, proposition,
corollary, claim, definition-like
implication, or assertion), decompose it
into:

1. hypotheses/assumptions
2. conclusions
Also rewrite the theorem in
normalized_form which must
be a single string of the form:
  "If [H1] and [H2] and ... then [C]."
```

The idea is that by explicitly stating the assumptions and conclusion, an LLM would be less likely to change the theorem in ways that made it mathematical different to rephrasing and similarly, it would be easier to control how an LLM changes a theorem to make it a hard negative.

Using the theorem decompositions, we gave these to `google/gemma-4-E4B-it` and requested rephrasings using the prompt:

```
You are a mathematical writer helping
to build a training dataset for an \
embedding model. Your task is to
produce alternative natural language \
phrasings of mathematical statements
that preserve their meaning exactly.

You are given the original statement
together with a pre-computed
decomposition into hypotheses and
conclusions. Use that decomposition
as your guide to reassemble the
components into a rephrasing with
a different surface form.

The rephrasing must be mathematically
identical to the original — it must \
make the same claims with the same
assumptions — but it should differ
in at least one of the following ways:
  - Sentence structure (e.g. "If A
  then B" → "Whenever A, we have B")
  - Logical connective style ("Let ...
  Then ..." → "Given ..., it follows ...")
  - Quantifier ordering or grouping
  - Passive vs active voice
  - Mathematical phrasing conventions
  (e.g. "there exists" → "we can find")
  - Naming convention (e.g. "a
  subgroup H" → "a subgroup, which we
  call H")
  - Consistently swap variables so
  "a + b = c" becomes "s + r = p" but
  be sure to keep conistency through
  the whole statement.
Overall, the phasing should be as
different as possible while remaining
completely mathematically equivalent.

The rephrasing should read as natural
mathematical prose — the kind you would \
find in a textbook or research paper.
Include LaTeX notation where the \
original does.

Output the rephrased statement
```

```
directly — no preamble, no explanation, \
no markup. Just the rephrasing itself.
"""

USER_TMPL = """\
Mathematical statement:
\"\"\"STATEMENT\"\"\"

Pre-computed decomposition:
Hypotheses:
HYPOTHESES
Conclusions:
CONCLUSIONS

Rephrasing:"""

RETRY_SUFFIX = (
    "\n\nYour previous response
    was empty. "
    "Output only the rephrased
    statement — no preamble
    or explanation."
)
```

Finally, we also prompted `Qwen/Qwen3.5-9B` to give us three hard negatives for the theorem using the prompt:

```
You are an expert mathematical editor.

Task:
Given a string with
"input_statement", "hypotheses",
"conclusions", "normalized_form",
"subject"), generate exactly 3
"hard_negatives" — statements
that closely mimic the visual and
syntactic structure of the
"input_statement" but are strictly
mathematically false or
semantically different.

Guidelines for Hard Negatives
(This is not an exhaustive list
of guidelines):
1. Subtle Omission: If it would
create a mathematically different
statement then omit a property that
is a prerequisite for the conclusion,
or remove a specific conclusion
2. Commutative Flip: If it would
create a mathematically different
statement then reverse the order of
a product that is not commutable in
a way that is plausible but strictly
breaks the equality.
3. Negation: If a negation would
create a mathematically different
statement, then negate an assumption
or conclusion to fundamentally alter
the semantic meaning of the statement.

Validity:
- Every variable used in the conclusion
must be defined in the hypotheses.
Every operator must act on the correct
type of object.
- The statement must be read as a
valid, grammatically correct mathematical
sentence, even though the underlying
claim is false.
- It cannot simply be a "weaker" true
statement. If the original theorem
guarantees x > 0, generating a negative
that says x \ge 0 is invalid because it
is still a true mathematical statement.
Do not merely omit a conclusion or
weaken an inequality without introducing
a strict contradiction or a false
mathematical guarantee.

Style Constraints:
- Preserve the original LaTeX formatting
style, notation, and complexity.
- The visual "weight" of each equation
must match the original theorem exactly.
```

C.2 The role of hard negatives

The hard negatives are meant to be mathematical statements with lexical similarity but completely different mathematical meaning. Often, this meant making resulting statement entirely false.

During initial fine-tuning runs, we attempted to use the hard negatives alongside the rephrasings for fine-tuning the text embedding models. However, we found that with our training techniques, fine-tuning with the hard negatives made the models worse. So, we decided to go with only the rephrasings. We leave the hard negatives as a potential future tool for different training techniques that might be able to make better use of them.

D License and Intended Use Information

Datasets used.

- `FrenzyMath/mathlib_informal_v4.19.0`: Apache 2.0. Used as the source of aligned mathlib4 statement views.
- `mathlib4` theorem statements (The mathlib Community, 2020): Apache 2.0.

Models used as fine-tuning bases.

- `Qwen/Qwen3-Embedding-8B`: Apache 2.0. Base for MathLeap-Qwen-8B.
- `Octen/Octen-Embedding-8B`: Apache 2.0. Base for MathLeap-Octen-8B.

Models used as evaluation baselines. The following models were used solely for evaluation; we did not fine-tune or redistribute them.

- `Qwen/Qwen3-Embedding-4B`: Apache 2.0.
- `microsoft/harrier-oss-v1-27b`: MIT license.
- `KaLM-Embedding-Gemma3-12B-2511`: subject to the Gemma license (inherited from the Gemma 3 base).
- `nvidia/llama-embed-nemotron-8b`: NVIDIA License, incorporating the Llama 3.1 Community License Agreement and the Llama 3.1 Acceptable Use Policy (built on `meta-llama/Llama-3.1-8B`).
- `F2LLM-v2-14B`: Apache 2.0.

LLMs used for data generation. We used the following models to generate rephrasings, hard negatives, and MELD statement pairs, in compliance with their terms of service.

- `Claude Opus 4.7` (Anthropic API): used for generating initial MELD statement pairs.
- `GPT-5.5` (OpenAI API): used for MELD validation.
- `Qwen3.5-9B`: used for statement decomposition and hard-negative generation.
- `gemma-4-E4B-it`: used for rephrasing generation (subject to the Gemma license).

Artifacts we release.

- *Math2Vec Embedding Dataset*: derivative of FrenzyMath; released under Apache 2.0, matching upstream.
- *Mathematically Equivalent but Lexically Different Pairs (MELD)*: original to this work; released under Apache 2.0.
- *MathLeap-Qwen-8B*: released under Apache 2.0, matching Qwen3-Embedding-8B.
- *MathLeap-Octen-8B*: released under Apache 2.0, matching Octen-Embedding-8B.

Consistency with intended use. Our use of existing artifacts is consistent with their declared intended use as research resources. FrenzyMath is released as a training resource for math NLP research; Qwen3-Embedding-8B and Octen-Embedding-8B are released as base models intended for fine-tuning; the baseline models we evaluate are released for research benchmarking under licenses that permit non-commercial research evaluation. The Gemma-based and Llama-based (Nemotron) baselines carry additional usage terms beyond Apache 2.0; we use them only for non-commercial research evaluation as their licenses permit.

Intended use of released artifacts. The Math2Vec Embedding Dataset, MELD benchmark, and MathLeap-Qwen-8B / MathLeap-Octen-8B are released for research in mathematical information retrieval and analysis of embedding model representations of mathematical content. They are not intended for safety-critical applications (e.g., automated theorem verification without human review) or for substantially out-of-distribution use beyond mathlib4-style content. MELD, in particular, should be used as a probe of embedding model behavior on cross-presentation equivalence rather than as ground truth for downstream mathematical correctness checking.