UniMath-RAG: A Unified Equation-Context Retrieval Framework for Enhancing LLM-based Mathematical Question Answering

Connor Cihula and Yiu-Kai Ng

Brigham Young University, Provo UT 84602, USA
connor.cihula@gmail.com, ng@compsci.byu.edu

Abstract. Mathematical question answering (Math QA) remains a challenging problem for large language models (LLMs) due to their limited symbolic reasoning and sensitivity to structural variations in equations. We propose **UniMath-RAG**, a unified retrieval-enhanced framework that integrates structure-aware equation matching with semantic embedding-based context retrieval. Unlike prior work that treats equations and text independently, our approach jointly models both via (i) a MathML-based unification algorithm capturing operator–operand structure and (ii) dense vector embeddings for contextual similarity. Retrieved question-answer pairs are used to condition and fine-tune LLMs. Extensive experiments on Math Stack Exchange and benchmark datasets demonstrate consistent improvements across P@K, MRR, and nDCG metrics, while ablation studies confirm the importance of symbolic structure and cross-dataset evaluations demonstrate strong generalization. Our results establish UniMath-RAG as a scalable and interpretable solution for integrating symbolic reasoning into data-driven AI systems.

Keywords: Math QA · Retrieval-Augmented Generation · Equation Matching · LLMs · Semantic Retrieval

1 Introduction

Mathematics plays a fundamental role across disciplines such as engineering, economics, computer science, and biology, and underpins advancements in areas including medicine and technology. It is also essential in everyday activities such as financial planning and scheduling. Despite its importance, mathematics is widely perceived as difficult, leading to reduced engagement [16]. This is particularly problematic in an era of increasing demand for technical literacy, as insufficient mathematical understanding can negatively impact both professional performance and everyday decision-making [8].

To improve accessibility, online Math Question Answering (Math QA) platforms have become widely used, enabling users to obtain answers from a large community of contributors [22]. While these systems provide broad coverage across topics and difficulty levels, they suffer from key limitations: responses

may be delayed, incomplete, or unavailable. Such constraints reduce their effectiveness, especially in time-sensitive scenarios.

A common approach to mitigating these issues is to recommend previously answered questions that are similar to a given query. However, the effectiveness of this approach depends critically on accurately capturing similarity between both mathematical expressions and their textual context. Existing retrieval methods primarily focus on natural language similarity and often fail to capture the structural properties of mathematical equations [18].

Meanwhile, Large Language Models (LLMs) have shown strong capabilities in language understanding, but remain limited in mathematical reasoning due to their sensitivity to symbolic structure and tendency to generate inconsistent or hallucinated outputs [12]. Although Retrieval-Augmented Generation (RAG) improves grounding [10], existing frameworks are not designed to handle the structural complexity of mathematical expressions. To address these challenges, we propose a unified Math QA framework that integrates *equation-level structural matching* with *semantic context similarity*. Specifically, we leverage *unification* to match equations based on their structure, and combine it with vector embeddings to capture textual similarity. This dual modeling approach enables more accurate retrieval of relevant question-answer pairs. Unlike prior methods that rely solely on symbolic or semantic similarity [4], our approach jointly considers both, allowing the system to identify deeper correspondences between problems even when surface representations differ. Furthermore, retrieved results are used to condition and fine-tune LLMs, improving their ability to generate accurate and consistent mathematical solutions.

The main contributions of this work are:

- *Structure-aware equation matching:* We introduce a MathML-based unification approach that captures hierarchical relationships among mathematical operators and operands.
- *Hybrid retrieval mechanism:* We combine symbolic matching with embedding-based similarity to improve retrieval accuracy.
- *LLM integration:* We demonstrate how retrieved question-answer pairs can be used to effectively condition and enhance LLM performance.
- *Comprehensive evaluation:* We validate the framework across multiple datasets and show consistent improvements over existing methods.

2 Related Work

A key challenge in Math QA is retrieving and comparing mathematical expressions, which exhibit hierarchical and symbolic structure beyond standard natural language. Structure-aware approaches represent equations using layout trees or graph-based models to capture operator-operand relationships and enable more accurate matching [15]. Systems such as MIaS further improve scalability through canonicalization and indexing of mathematical expressions [18], while benchmark efforts like ARQMath highlight the importance of combining textual and formula similarity for effective retrieval [22]. However, these methods typically emphasize either structural matching or textual similarity, limiting their ability to capture deeper equivalence between math-related expressions.

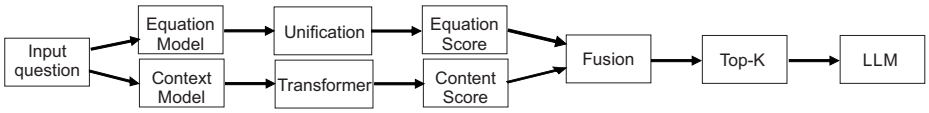

Fig. 1: UniMath-RAG pipeline architecture

Recent advances in representation learning have enabled the use of dense embeddings for retrieval. Transformer-based models such as Sentence-BERT [17] and MiniLM [20], combined with dense retrieval frameworks [9] and efficient indexing methods like FAISS [7], support scalable similarity search over large corpora. While effective for textual content, these approaches generally treat mathematical expressions as plain text, overlooking structural information essential for mathematical reasoning.

Large Language Models (LLMs) have further advanced question answering, yet remain limited in mathematical reasoning due to challenges in symbolic manipulation and multi-step inference. Models such as Minerva [12] and GPT-4, along with techniques like chain-of-thought prompting [21], improve performance but still exhibit inconsistency. Retrieval-Augmented Generation (RAG) [11] improves grounding and has been applied to math QA [10], but does not explicitly model equation structure. In contrast, our approach integrates structure-aware equation matching with embedding-based retrieval in a unified framework, enabling more accurate and robust mathematical reasoning.

3 The Proposed Math QA Model

Our Math QA framework, illustrated in Figure 1, computes similarity by jointly modeling equation structure and textual context. Given an input query, the system decomposes it into a structural representation of mathematical expressions and a semantic representation of surrounding text. Equation similarity is computed using a unification-based approach, where operators and operands are matched through a structured comparison of MathML representations. Specifically, each equation is transformed into a hierarchical MathML tree and traversed to produce array-based representations that facilitate efficient comparison. A scoring scheme then evaluates the degree of structural alignment between equations, producing a normalized similarity score in $[0, 1]$. In parallel, semantic similarity is computed using vector embeddings of the textual content. These two signals are subsequently fused to retrieve the most relevant question-answer pairs, which are used to enhance downstream LLM-based answer generation. This unified design enables our model to capture both symbolic equivalence and contextual relevance, addressing key limitations of existing Math QA approaches.

3.1 Structure-Aware Equation Matching via MathML Unification and Textual Content Analysis

Our tree matching approach improves Math QA accuracy by computing equation similarity through structure-aware matching of mathematical operators and operands, a core principle of *unification*. In this context, “like" elements refer to

compatible components, such as variables with variables or constants with constants, that can be aligned across expressions. The matching process is formalized using *unification*, a well-established technique in logic programming and theorem proving, which determines whether two symbolic expressions can be made equivalent through consistent substitutions. We adapt this concept to Math QA by representing equations in MathML and comparing their hierarchical structures to identify unifiable elements. The degree of alignment between operators and operands directly influences the similarity score, enabling the system to assess how closely one equation corresponds to another. This formulation allows the unification process to serve as an effective mechanism for retrieving relevant answers based on structural equivalence of mathematical expressions.

While equation matching captures structural similarity, it is insufficient for fully determining the relevance of candidate answers, as mathematically similar expressions may arise in different contexts or problem settings. To address this limitation, we incorporate semantic analysis of the textual content associated with math questions and answers, enabling the system to account for intent, problem description, and contextual nuances. Specifically, we employ a similarity search framework based on the all-MiniLM-L6-v2 sentence transformer combined with Facebook AI Similarity Search (FAISS) to retrieve semantically relevant question–answer pairs from Math Stack Exchange (MSE). This content-based retrieval complements equation matching by identifying answers that are not only structurally similar but also contextually appropriate. The retrieved pairs are subsequently used to fine-tune a large language model (LLM), whose performance is evaluated against a baseline model without fine-tuning. This comparison allows us to assess the effectiveness of integrating structural and semantic retrieval for improving Math QA accuracy.

3.2 Tree Traversal and Path-Based Unification

Our Math QA algorithm operationalizes unification through a structured comparison of MathML trees derived from the query and candidate answers. Specifically, each equation is parsed using Python's ElementTree module and traversed via depth-first search (DFS) to produce an array of root-to-leaf paths, where each subarray corresponds to a branch in the MathML tree. These path representations, referred to as the Right Tree (query) and Left Tree (candidate), encode the hierarchical structure of the equation while enabling efficient comparison in array form. To preserve structural distinctions, numeric identifiers are injected into paths to differentiate roles such as numerator versus denominator, base versus exponent, and related positional relationships (see Figure 2a and 2b for examples).

During matching, unification is applied at the tag level rather than the token level, i.e., tags representing variables (e.g., <mi>) are matched with other variable tags, similarly for operators and constants, while specific symbol identities are ignored (see Table 1 for the list of MathML tags used for path-based unification). This abstraction allows structurally equivalent expressions (e.g., $a + b$ and $c + d$) to be recognized as similar, focusing the comparison on structural alignment rather than superficial symbol differences.

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
 <mrow>
  <mfrac>
   <mrow>
    <msqrt>
     <mrow>
      <msup>
       <mi>a</mi> (8)
       <mn>2</mn> (7)
      </msup>
      <mo>-</mo> (6)
      <mi>a</mi> (5)
     </mrow>
    </msqrt>
   </mrow>
   <mrow>
    <mi>b</mi> (4)
   </mrow>
  </mfrac>
  <mo>+</mo> (3)
  <msup>
   <mi>c</mi> (2)
   <mn>4</mn> (1)
  </msup>
 </mrow>
</math>
```

(a) MathML tree representation of the query equation, $\frac{\sqrt{a^2-a}}{b} + c^4$

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
 <mrow>
  <mfrac>
   <mrow>
    <msup>
     <mi>x</mi> (7)
     <mn>2</mn> (6)
    </msup>
    <mo>+</mo> (5)
    <mi>y</mi> (4)
   </mrow>
   <mrow>
    <mi>z</mi> (3)
   </mrow>
  </mfrac>
  <mo>-</mo> (2)
  <mi>f</mi> (1)
 </mrow>
</math>
```

(b) MathML tree representation of the candidate answer, $\frac{x^2+y}{z} - f$

Fig. 2: Example of tree-based representations used in path-based unification. Each equation is converted into a MathML tree and traversed for comparison

The resulting path-based representation provides a robust and computationally efficient foundation for measuring equation similarity. Figures 3a and 3b illustrate this process using two structurally similar equations. Each equation is transformed into a set of root-to-leaf paths, where corresponding branches encode analogous structural roles (e.g., operand positions within an addition or multiplication). Although the specific variables differ, the paths terminate in identical tag sequences, allowing the algorithm to align branches across the two trees. For example, the paths representing the left operand of an addition in both equations share the same tag pattern, enabling them to be matched despite differing symbols. As a result, structurally equivalent expressions such as $a + b$ and $c + d$ yield high similarity scores, demonstrating that our approach captures structural equivalence rather than superficial symbol matching. This path-based representation provides a robust and computationally efficient foundation for measuring equation similarity in Math QA.

We quantify the similarity between two equations by computing a score that reflects the degree of alignment between their respective branches. This overall similarity score is derived from two components: a *depth_score*, which measures structural alignment at the branch level, and a *complexity_score*, which accounts for differences in equation size.

Table 1: List of all MathML tags

Tags	Interpretation
<math>	Top level element, indicating the start of a formula
<mfrac>	Branches into two children, numerator & denominator
<mrow>	Groups items together, such as parentheses
<msqrt>	Square root
<mroot>	Branches into two children, the root and its degree
<msub>	Branches into two children, the base and its subscript
<msup>	Same as above except second child is superscript
<msubsup>	Branches into three children, the base, its subscript, and its superscript
<munder>	Branches into two children, the base and under
<mover>	Branches into two children, the base and over
<munderover>	Branches into three children, the base, under, and over
<mn>	Wraps a single literal (decimal or integer)
<mi>	Wraps a single "identifier", i.e., variable, function (cos, sin, etc.), and symbol (Π, θ, etc.))
<mo>	Wraps any single operator (+, -, etc.)
<mtext>	Wraps text (if, maps to, etc.)

Depth_Score The depth_score evaluates structural similarity by comparing each branch in the Right Tree (query) with all branches in the Left Tree (candidate) and selecting the best match. For each pair of branches, we apply a dynamic programming algorithm to compute the length of the longest contiguous subsequence of MathML tags shared by both branches. This effectively captures the extent of local structural alignment. The root `<math>` tag is excluded from the comparison, as it appears in all expressions and does not contribute to discriminative similarity.

Example 1. Consider Figures 3a and 3b. Branch 4 of the Right Tree matches branch 5 of the Left Tree with a score of $\frac{4}{5}$, where 4 denotes the length of the longest matching subsequence (e.g., `mrow`, `mfrac`, 0, `mrow`) and 5 is the total number of tags in the Right Tree branch (excluding the root). This process is repeated for all branches in the Right Tree, yielding a set of branch-level scores that collectively characterize the structural similarity between the two equations. □

1. [math, mrow, msup, 1, mn], (4)
2. [math, mrow, msup, 0, mi], (c)
3. [math, mrow, mo], (+)
4. [math, mrow, mfrac, 1, mrow, mi], (b)
5. [math, mrow, mfrac, 0, mrow, msqrt, mrow, mi], (a)
6. [math, mrow, mfrac, 0, mrow, msqrt, mrow, mo], (-)
7. [math, mrow, mfrac, 0, mrow, msqrt, mrow, msup, 1, mn], (2)
8. [math, mrow, mfrac, 0, mrow, msqrt, mrow, msup, 0, mi], (a)

(a) The branch array of the equation $\frac{\sqrt{a^2-a}}{b} + c^4$ as shown in Figure 2a

1. [math, mrow, mi], (f)
2. [math, mrow, mo], (-)
3. [math, mrow, mfrac, 1, mrow, mi], (z)
4. [math, mrow, mfrac, 0, mrow, mi], (y)
5. [math, mrow, mfrac, 0, mrow, mo], (+)
6. [math, mrow, mfrac, 0, mrow, msup, 1, mn], (2)
7. [math, mrow, mfrac, 0, mrow, msup, 0, mi], (x)

(b) The branch array of the equation $\frac{x^2+y}{z} - f$ as shown in Figure 2b

Fig. 3: Branch arrays of the query and candidate equations across the two different trees representing two equations

Complexity_Score The complexity_score complements the depth_score by accounting for differences in structural size between two equations. It is computed as a normalized ratio of the number of MathML tags in the Right and Left Trees as shown below.

$$complexity\_score = \frac{min(LTT, RTT)}{max(LTT, RTT)} \tag{1}$$

which yields a value in $[0, 1]$, where 1 indicates equal structural complexity and lower values reflect increasing disparity.

This component mitigates a key limitation of the depth_score: since multiple branches in the Right Tree may align with a single branch in the Left Tree, depth-based matching alone can overestimate similarity. For instance, a simple expression such as $\frac{2}{3}$ may achieve a high depth_score when compared to a substantially more complex expression if a few structural patterns coincide. While the depth_score captures shared structure, it does not penalize unmatched components, leading to inflated similarity estimates.

The complexity_score addresses this issue by penalizing such imbalances, ensuring that large structural differences reduce the overall similarity. Together, the depth_score and complexity_score provide a balanced measure that captures both structural alignment and relative complexity. Compared to bidirectional matching strategies, which would require additional depth computations, the proposed formulation achieves comparable accuracy with lower computational cost, preserving the efficiency of the framework.

The Equation Similarity Score The equation similarity score, denoted EQS_Score and as defined below in which n denotes the number of *branches* in $Query\_Tree$, combines the two scores through a depth-adaptive weighting scheme.

$$EQS\_Score = Complexity\_Score \times \sum_{i=1}^{n} \frac{Depth\_Score_i}{n} \tag{2}$$

Specifically, the contribution of the complexity_score increases as the depth_score grows, reflecting its role as a corrective factor for potentially inflated structural matches. When the depth_core is low, the overall similarity remains dominated by structural mismatch, and the impact of the complexity_core is correspondingly limited. Conversely, when the depth_score is high, the complexity_score exerts greater influence, penalizing discrepancies in equation size and preventing overestimation of similarity. This formulation ensures that the similarity score balances structural alignment and relative complexity, while remaining bounded in $[0, 1]$, where higher values indicate greater similarity.

3.3 The Content Similarity Score

To enable efficient content-based retrieval, we employ a similarity search framework that combines the all-MiniLM-L6-v2 sentence transformer with Facebook AI Similarity Search (FAISS). The transformer model generates dense vector embeddings for both textual content and MathML-encoded equations, capturing contextual and semantic information in a high-dimensional space. Similarity

between queries and candidate answers is then computed using cosine similarity, where semantically related items correspond to vectors with small angular distance.

all-MiniLM-L6-Ve. Although originally designed for natural language, all-MiniLM-L6-v2 can be effectively applied to MathML representations by treating them as structured text sequences. This allows the model to capture contextual relationships among mathematical components with minimal loss of meaning. By embedding both natural language descriptions and equation representations into a unified vector space, the framework supports consistent and accurate similarity comparison across modalities. Coupled with FAISS for scalable nearest-neighbor search, this approach enables fast retrieval over large QA corpora while maintaining strong semantic fidelity.

FAISS. After transforming textual content and written text in MathML equations into vector embeddings using the all-MiniLM-L6-v2 model, we index the resulting representations with FAISS to enable efficient similarity search over the chosen Math Stack Exchange (MSE) corpus. FAISS supports scalable nearest-neighbor retrieval by organizing high-dimensional vectors into optimized index structures, leveraging techniques such as product quantization and clustering to reduce computational cost while preserving approximate similarity relationships.

To maximize efficiency, we adopt a two-stage retrieval strategy. Rather than indexing all answers directly, which would be computationally expensive given that each question may have multiple associated answers, we first construct an index over MSE questions. For a given query, the system retrieves the top-k most similar questions from this index. A secondary FAISS index is then built from the answers associated with these retrieved questions, and the query is compared against this reduced candidate set to identify the most relevant answers. This hierarchical approach balances scalability and accuracy by limiting the search space while retaining high-quality candidates. Similarity is computed using distances between both semantic text embeddings and equation embeddings, ensuring that retrieved results reflect both contextual and structural relevance.

3.4 The Combined Structured and Content Similarity Score

The equation similarity score (EQS_Score) and the content similarity score (CS_Score) for a math question A and a candidate answer Q are combined using the *Borda Count strategy* [3] to produce a final ranking score. The Borda Count is a rank-based aggregation method that combines multiple ranked lists by assigning scores based on the relative positions of items rather than their raw similarity values. This makes it particularly suitable for our setting, where structural and semantic similarity scores may have different scales and distributions.

Let $\mathcal{Q} = \{Q_1, Q_2, \ldots, Q_n\}$ denote the set of candidate answers for a query A. Each candidate Q_i $(1 \leq i \leq n)$ is independently ranked according to EQS_Score and CS_Score, producing two ranked lists. For a given ranking, the Borda score assigns $n - r_i(Q_j)$ points to candidate Q_j, where $r_i(Q_j)$ is the rank position of Q_j under scoring function $i \in \{\text{EQS}, \text{CS}\}$, and n is the total number of candidates. Higher-ranked candidates receive more points.

The final aggregated score for each candidate Q_j is computed as:

$$textBorda(Q_j) = \sum_{i \in \{\text{EQS,CS}\}} \big(n - r_i(Q_j)\big) \tag{3}$$

Candidates are then ranked in descending order of their Borda scores to produce the final list of recommended answers.

This rank-based aggregation provides two key advantages. First, it reduces the influence of extreme scores from either similarity measure, ensuring that no single signal dominates the ranking. Second, it captures complementary information from both structural (equation-level) and semantic (content-level) similarity, leading to more robust and balanced retrieval results. The resulting ranking reflects overall consensus between the two similarity measures rather than reliance on any single metric.

3.5 Fine-Tuning LLMs Using Our Math QA Math

Fine-tuning adapts a pre-trained language model to a target domain by updating its parameters using task-specific data, leveraging its existing linguistic and reasoning capabilities while avoiding the cost of training from scratch. In the context of mathematical question answering (Math QA), fine-tuning is particularly valuable for aligning the model with domain-specific structures, terminology, and reasoning patterns that are not fully captured during general pretraining.

In this work, we explore a *retrieval-augmented fine-tuning* strategy that leverages our similarity search framework to provide high-quality, contextually relevant supervision signals. Rather than relying on randomly sampled training data, we construct fine-tuning inputs using question–answer pairs retrieved based on both structural (equation-level) and semantic similarity. This ensures that the model is exposed to examples that closely match the query distribution, thereby improving its ability to generalize to mathematically similar problems.

We evaluate this approach using three compact yet capable open-weight models: LLaMA3.1-8B, Qwen2.5-7B, and Gemma3-4B. These models are selected to balance reasoning capability with computational efficiency, making them suitable for controlled experimentation in resource-constrained settings. While larger models may achieve higher absolute performance, our focus is on demonstrating that retrieval-informed fine-tuning can significantly enhance the capabilities of moderate-scale models.

To assess effectiveness, we adopt a controlled evaluation protocol. Each model is first evaluated on a held-out subset of Math Stack Exchange (MSE) questions to establish a baseline. We then augment the model's input using retrieved question–answer pairs drawn from a disjoint subset of MSE data, ensuring no data leakage. This augmentation can be viewed as a lightweight form of fine-tuning or in-context adaptation, where the model conditions on relevant examples at inference time. The only difference between the baseline and enhanced settings is the inclusion of retrieval-based context, isolating the impact of our method.

Our results show that incorporating structurally and semantically aligned examples leads to consistent improvements in answer accuracy, coherence, and

mathematical correctness. Notably, these gains are achieved without modifying model architecture or requiring large-scale retraining. This demonstrates the effectiveness of combining retrieval and fine-tuning signals, and highlights the potential of retrieval-augmented approaches as a practical and scalable pathway for improving LLM performance on structured reasoning tasks such as Math QA.

Language Models for Retrieval-Augmented Adaptation To evaluate the effectiveness of our retrieval-augmented fine-tuning strategy, we select LLaMA, Qwen, and Gemma, three open-weight LLMs that represent modern mid-scale architectures balancing reasoning capability and computational efficiency. They provide a controlled testbed for assessing whether retrieval-informed adaptation can consistently improve mathematical reasoning across diverse model designs.

- **LLaMA3.1-8B (Meta).**
  A strong general-purpose model with robust instruction-following and reasoning capabilities, serving as a reliable baseline for domain-specific adaptation in Math QA.
- **Qwen2.5-7B (Alibaba).** A recent model incorporating improved alignment and training strategies, particularly effective for structured reasoning and multilingual tasks, providing architectural diversity for evaluating generalization.
- **Gemma3-4B (Google DeepMind).** A lightweight model optimized for efficient inference and rapid adaptation, enabling analysis of retrieval gains under constrained model capacity.

Retrieval-Augmented Fine-Tuning Protocol We adopt a controlled evaluation protocol to isolate the impact of retrieval-based adaptation. Each model is first evaluated on a held-out subset of Math Stack Exchange (MSE) questions to establish a baseline without domain-specific augmentation. We then enhance the model inputs using question–answer pairs retrieved by our similarity framework from a disjoint subset of MSE data, ensuring no overlap with the evaluation set.

This retrieval-driven augmentation serves as a targeted fine-tuning signal, exposing the model to structurally and semantically aligned examples. Unlike standard fine-tuning, which relies on static datasets, our approach dynamically selects context based on query similarity, effectively bridging retrieval and generation. Performance differences between baseline and augmented settings are therefore attributable solely to the inclusion of retrieval-informed context.

Across all models, we observe consistent improvements in answer accuracy, coherence, and mathematical validity (see Section 4). These results demonstrate that integrating structured (equation-level) and semantic (content-level) retrieval signals provides an effective and scalable mechanism for enhancing LLM performance on Math QA, even when training and evaluation data are disjoint.

4 Experimental Results

This section evaluates the effectiveness of the proposed Math QA-LLM pipeline in solving mathematical problems that require integrated reasoning over natural language, symbolic equations, and visual representations. Our primary objective is threefold: (i) to assess whether the proposed pipeline improves answer accuracy

and reasoning quality, (ii) to benchmark its performance against both strong baseline LLMs and existing Math QA systems, and (iii) to validate its ability to generalize across diverse mathematical domains and problem formats, ranging from grade-school word problems to competition-level and multi-domain STEM questions. By explicitly comparing against prior approaches, we aim to determine whether cross-modal integration and structured mathematical representations translate into more reliable reasoning and precise final answers.

To ensure a rigorous and comprehensive evaluation, we first introduce the datasets and experimental setup used to train and test the model, followed by the evaluation protocol that governs how outputs are assessed. We begin by evaluating the proposed pipeline as a performance-enhancing framework for modern LLMs, examining whether its integration can systematically improve reasoning accuracy, solution completeness, and final answer precision. In particular, we study its impact on representative models, i.e., LLaMA 3.1-8B, Qwen 2.5-7B, and Gemma 3-4B, focusing on gains achieved through retrieval-augmented reasoning, structured mathematical grounding, and cross-modal alignment.

Building on this analysis, we then compare our approach with established Math QA systems, including *MaRec* [5], *ProblemSolver* [14], *AiFu system* [13], and *MIaS* [18] system. These systems serve as task-specific baselines, enabling us to evaluate how the proposed pipeline advances beyond prior methods that typically emphasize isolated components such as retrieval, symbolic manipulation, or text-based reasoning. This two-stage evaluation—first enhancing LLMs and then benchmarking against specialized Math QA systems—provides a comprehensive validation of the proposed framework's effectiveness across both general-purpose and domain-specific settings.

4.1 Datasets and Evaluation Protocol

To evaluate the proposed Math QA model—designed to jointly process natural language, mathematical expressions, and visual representations—we construct a specialized text-equation dataset derived from the *Math Stack Exchange* (MSE) dataset. This curated subset contains 64,860 questions paired with 730,941 answers (≈10.8 answers per question), along with metadata such as answer scores, acceptance indicators, and rendered image versions of both questions and answers generated from LaTeX markup.

The dataset is partitioned into 90% training, 5% validation, and 5% testing. The training split is used to learn a unified Math QA model capable of ingesting textual descriptions, symbolic equations, and visualized mathematical content, and producing answers across these modalities. This design supports richer reasoning and more interpretable responses, enabling learners to engage with solutions presented in multiple complementary forms.

4.2 Benchmark Datasets for Generalization

To further assess the effectiveness and generalization of the proposed Math QA-LLM pipeline, we evaluate it against several widely used benchmarks and compare performance with strong baseline models, including LLaMA 3.1-8B, Qwen 2.5-7B, and Gemma 3-4B. The evaluation spans diverse mathematical tasks using the following datasets:

(i) MATH dataset [6]: Comprising 12,500 challenging high school competition problems, this dataset provides detailed, step-by-step solutions, making it well-suited for evaluating reasoning, derivation, and explanation generation.
(ii) GSM8K [2]: A collection of 8.5K linguistically diverse word problems focused on elementary-level mathematics, designed to test reasoning under varied natural language formulations.
(iii) SAT May 2023 Math Section [1]: This dataset includes 32 exam-style math questions (without diagrams), reflecting standardized testing scenarios.
(iv) MMLU-STEM [6]: Containing approximately 3,150 questions, this benchmark spans multiple STEM domains and evaluates both factual knowledge and problem-solving across a wide difficulty range.

4.3 Evaluation Setup and Answer Verification

All models—both the proposed pipeline and baselines—are evaluated on uniformly structured benchmark datasets, each consisting of question-answer (QA) pairs. Despite covering different domains (e.g., competition math, word problems, standardized exams, and general STEM knowledge), these datasets share a consistent format:

(i) Each question is paired with a single validated reference answer.
(ii) Both questions and answers include a mix of natural language explanations and mathematical expressions.
(iii) Solutions typically present step-by-step reasoning, followed by a clearly stated final answer in mathematical notation.

Model outputs are assessed using exact-match evaluation, where the generated final answer is compared directly against the ground-truth answer. This ensures precise verification of correctness, particularly for problems where the final numeric or symbolic result is critical.

4.4 Performance Evaluation Using Math QA-LLM Pipeline

Based on the performance metrics reported in Table 2, the results demonstrate that the pipeline introduces a novel retrieval-guided fine-tuning paradigm that substantially improves mathematical question answering for medium- and large-scale LLMs. In particular, LLaMA 3.1-8B exhibits the most pronounced gains, with notable improvements across nearly all evaluation metrics—especially a sharp increase in AP, indicating stronger prioritization of relevant answers. Similarly, Qwen 2.5-7B shows consistent, albeit more moderate, improvements across most metrics. These findings confirm that integrating similarity-based retrieval into the fine-tuning process enhances both answer relevance and ranking quality, highlighting the pipeline's ability to guide LLM reasoning through structured, context-aware retrieval signals. Taken together, these results highlight both the strength and adaptability of the MathQA-LLM framework: it not only delivers measurable performance improvements for capable LLMs, but also exposes critical factors—such as model size, architecture, and training dynamics—that inform future optimization. Consequently, the proposed pipeline establishes a scalable and extensible foundation for advancing mathematical reasoning in LLMs while motivating targeted refinements for broader applicability.

Table 2: LLM Performance with(out) Fine-Tuned Using our Math QA System

Model	Fine-tuned	P@3	P@5	AP	RR	RR@5	nDCG@1	nDCG@5
LLaMA 3.1-8B	No	0.633	0.560	0.611	0.683	0.683	0.695	0.707
	Yes	0.633	0.580↑	0.689↑	0.750↑	0.750↑	0.774↑	0.785↑
Qwen 2.5-7B	No	0.833	0.780	0.868	1.000	1.000	0.950	0.975
	Yes	0.967↑	0.900↑	0.943↑	0.950↓	0.950↓	0.972↑	0.976↑
Gemma 3-4B	No	0.600	0.640	0.681	0.817	0.800	0.777	0.757
	Yes	0.633↑	0.650↑	0.612↓	0.850↑	0.650↓	0.786↑	0.668↓

To validate the effectiveness of the proposed MathQA system, we apply the Wilcoxon Signed-Rank Test [19] to assess whether performance gains over baseline models are statistically significant. As shown in Table 3, the proposed model consistently outperforms both fine-tuned and non-fine-tuned versions of LLaMA 3.1-8B, Qwen 2.5-7B, and Gemma 3-4B across most evaluation metrics.

Table 3: Wilcoxon Signed-Rank Test P-Values (Compared to our MathQA Tool)

Comparison	P@3	P@5	AP	RR	RR@5	nDCG@1	nDCG@5
LLaMA (w/o)	0.0026	0.0377	0.0007	0.0006	0.0577	0.0432	0.0286
LLaMA (w/)	0.00006	0.000002	0.00000002	0.0000006	0.7718	0.0094	0.0578
Qwen (w/o)	0.03426	0.0500	0.0126	0.1192	0.0175	0.7125	0.0471
Qwen (w/)	0.000005	0.0001	0.0000005	0.00002	0.0000009	0.7006	0.0654
Gemma (w/o)	0.0001	0.0118	0.0002	0.00002	0.0547	0.03283	0.0359
Gemma (w/)	0.005	0.00003	0.003	0.002	0.0030	0.0000008	0.08923

The majority of p-values are below 0.05, with many significantly smaller, confirming that the improvements are statistically robust. In particular, gains in precision (e.g., $P@3$, $P@5$) and ranking metrics (AP, RR, $nDCG$) indicate more accurate and consistent answer ranking. Although a few cases do not reach significance, the overall pattern demonstrates that the proposed MathQA system achieves reliable and meaningful performance improvements, validating the effectiveness of its retrieval-guided, cross-modal design.

4.5 Comparing Performance Evaluation of Various Math QA Models

To rigorously assess the effectiveness of the proposed Math QA system, we conduct a comparative evaluation against several established approaches, including MaRec, ProblemSolver, AiFu system, and MIaS system. Unlike prior systems that typically emphasize isolated capabilities—such as retrieval (MIaS), symbolic manipulation (ProblemSolver), or pattern-based answer generation (MaRec and AiFu)—our model introduces a unified framework that jointly integrates retrieval, reasoning, and cross-modal understanding of text, equations, and visual representations. This design enables the system to move beyond traditional keyword or structure-driven matching toward context-aware reasoning and answer synthesis, thereby addressing key limitations in existing Math QA approaches.

We evaluate all five systems using standard information retrieval metrics that reflect real user behavior, including $P@1$, $P@5$, MRR, and $NDCG@5$, where emphasis is placed on the quality of top-ranked results. As illustrated in Figure 4,

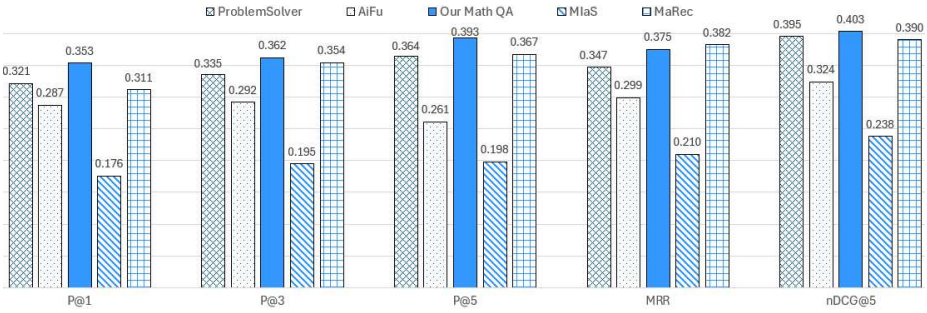

Fig. 4: The performance evaluation metrics of Math QA systems, including ours

the proposed Math QA model consistently outperforms all baselines across every metric. In particular, higher $P@1$ and $P@5$ scores indicate that our system more effectively ranks correct and relevant answers at the top positions, while improvements in MRR and $NDCG@5$ demonstrate superior ranking quality and robustness across candidate answers. These gains are statistically significant under the Wilcoxon Signed-Rank test ($p < 0.045$), confirming that the observed improvements are not due to random variation.

4.6 Question Processing Time

A design goal of the proposed Math QA-LLM pipeline is to achieve question processing times comparable to existing Web search engines. To evaluate efficiency, we measured the average processing time for retrieving and ranking answers to math questions in the MSE dataset against baseline Math QA systems, including MaRec, PowerSolver, MIaS, AiFu, and ChatGPT. All experiments were conducted on the same MacBook Pro with an M1 chip. As shown in Table 4, the proposed model runs nearly twice as fast as ChatGPT, although it remains slower than specialized Math QA baselines. These results demonstrate that the proposed pipeline improves the practicality of LLM-based Math QA systems.

Table 4: Question processing time (in seconds) between $MaRec$, $ProblemSolver$, $MIaS$, $AiFu$, the proposed pipeline model, and $ChatGPT$

Models/	MaRec	Problem Solver	MIaS	AiFu	Our Pipeline Model	ChatGPT
Processing Time	0.037s	0.229s	2.05s	4.000s	7.912s	15.042s

5 Conclusion

This work presents a MathQA-LLM pipeline that advances mathematical question answering by unifying retrieval, reasoning, and cross-modal understanding within a single framework. By tightly integrating similarity-guided retrieval with structured mathematical representations, the proposed system moves beyond prior approaches that treat retrieval, symbolic processing, and answer generation independently, resulting in more accurate, context-aware, and interpretable solutions. Empirical results demonstrate statistically significant improvements over both general-purpose LLMs and established Math QA systems, confirming the robustness and effectiveness of the approach across diverse math tasks.

References

1. Azerbayev, Z., et al.: Llemma: An Open Language Model for Mathematics.arXiv preprint arXiv:2310.10631 (2023)
2. Cobbe, K., et al.: Training Verifiers to Solve Math Word Problems. arXiv preprint arXiv:2110.14168 (2021)
3. Fraenkel, J., Grofman, B.: The Borda Count and Its Real-World Alternatives: Comparing Scoring Rules in Nauru and Slovenia. AJPS **49**(2), 186–205 (2014)
4. Fritz, A., Haase, V., Rasanen, P.: International Handbook of Mathematical Learning Difficulties. Cham, Switzerland: Springer (2019)
5. Gao, S., Ng, Y.: Recommending Answers to Math Questions Based on KL-Divergence and Approximate XML Tree Matching. In: Proc. of ACM SIGIR-AP. pp. 21–31 (2023)
6. Hendrycks, D., et al.: Measuring Mathematical Problem Solving with the Math Dataset. In: Proc. of the NIPS Track on Datasets and Benchmarks (2021)
7. Johnson, J., Douze, M., Jégou, H.: Billion-Scale Similarity Search with GPUs. IEEE Transactions on Big Data **7**(3), 535–547 (2019)
8. Kappassova, S., et al.: Mathematical Literacy and Its Influencing Factors: A Decade of Rresearch Findings (2015-2024). EJMSTE **21**(7), em2671 (2025)
9. Karpukhin, V., et al.: Dense Passage Retrieval for Open-Domain Question Answering. In: Proc. of EMNLP. pp. 6769–6781 (2020)
10. Levonian, Z., et al.: Retrieval-Augmented Generation to Improve Math Question-Answering: Trade-Offs between Groundedness and Human Preference. arXiv preprint arXiv:2310.03184 (2023)
11. Lewis, P., et al.: Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. NeurIPS **33**, 9459–9474 (2020)
12. Lewkowycz, A., et al.: Solving Quantitative Reasoning Problems with Language Models. Advances in Neural Information Processing Systems **35**, 3843–3857 (2022)
13. Liu, Y., Ding, K., Zhou, Y.: Aifu at semeval-2019 task 10: A symbolic and sub-symbolic integrated system for sat math question answering. In: Proc. of the 13th Workshop on Semantic Evaluation. pp. 900–906 (2019)
14. Luo, X., Baranova, A., Biegert, J.: Problemsolver at Semeval-2019 Task 10: Sequence-to-Sequence Learning and Expression trees. In: Proc. of the 13th Workshop on SemEval. pp. 1292–1296 (2019)
15. Mansouri, B., et al.: Tangent-CFT: An Embedding Model for Mathematical Formulas. In: Proc. of ACM SIGIR. pp. 11–18 (2019)
16. Rattan, A., Good, C., Dweck, C.: 'It's OK — Not Everyone can be Good at Math': Instructors with an Entity Theory Comfort (and Demotivate) Students. Experimental Social Psychology **48**(3), 731–737 (2012)
17. Reimers, N., Gurevych, I.: Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks. In: Proc. of EMNLP-IJCNLP. pp. 3982–3992 (2019)
18. Sojka, P., Ruzicka, M., Novotny, V.: MIaS: Math-Aware Retrieval in Digital Mathematical Libraries. In: Proc. of ACM CIKM. pp. 1923–1926 (2018)
19. Taheri, S., Hesamian, G.: A Generalization of the Wilcoxon Signed-Rank Test and Its Applications. Statistical Papers **54**(2), 457–470 (2013)
20. Wang, W., et al.: Minilm: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. NeuIPS **33**, 5776–5788 (2020)
21. Wei, J., et al.: Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. NeurIPS **35**, 24824–24837 (2022)
22. Zanibbi, R., Oard, D., Agarwal, A., Mansouri, B.: Overview of ARQMath: CLEF Lab on Answer Retrieval for Questions on Math. In: CLEF. pp. 169–193 (2020)