Beyond Accuracy: Alignment and Error Detection across Languages in the Bi-GSM8K Math-Teaching Benchmark

Jieun Park[†,*], **KyungTae Lim**[‡,*], **Joon-Ho Lim**[†]
[†]Tutorus Labs (AI Lab) [‡]Korea Advanced Institute of Science and Technology
{geeag51,jhlim}@tutoruslabs.com, ktlim@kaist.ac.kr

Abstract

Recent advancements in LLMs have significantly improved mathematical problem-solving, with models like GPT-4 achieving human-level performance. However, proficiently solving mathematical problems differs fundamentally from effectively teaching mathematics. To bridge this gap, we introduce the Bi-GSM8K benchmark, a bilingual English-Korean dataset enriched with teacher solutions, student solutions, and annotations marking students' initial errors. This dataset is designed to evaluate two core capabilities of LLMs: (1) measuring similarity between student and teacher solutions, and (2) identifying the initial error point in student solutions. Our method achieves high agreement with human judgments, with Pearson 0.89 and Spearman 0.88 on English, and Pearson 0.89 and Spearman 0.87 on Korean. It also offers significantly lower latency and resource usage than commercial APIs, demonstrating strong computational efficiency. In the error detection task, open-source models achieved approximately 86% accuracy, with performance within 10% points of commercial LLMs API, suggesting strong practical potential. Our key contributions include the open-source release of Bi-GSM8K, novel evaluation metrics, and comparative analyses of LLM performance across languages.

1 Introduction

Recent advancements in LLMs have led to significant progress in mathematical problem-solving tasks. Notably, GPT-4 has achieved accuracy rates of 97% and 86% on GSM8K and MMLU benchmarks, respectively, demonstrating performance comparable to expert human levels. Additionally, OpenAI's o1 model has attained an accuracy of 74.4% (pass@1) on the AIME problems, further evidencing its advanced reasoning capabilities (Zhong et al., 2024; Achiam et al., 2023; OpenAI, 2024b). These results indicate that LLMs have evolved from mere language-understanding tools into sophisticated instruments capable of logical reasoning and computational problem-solving.

However, effectively solving mathematical problems and proficiently teaching mathematics to students constitute fundamentally distinct tasks. Prior research has established that the competencies involved in effectively solving mathematical problems, termed Common Content Knowledge (CCK), differ significantly from the Mathematical Knowledge for Teaching (MKT) required for effective pedagogical practice (Understand, 1986; Ball et al., 2008). Within educational contexts, this implies that merely providing correct answers is insufficient; it is crucial to understand the student's thought processes and diagnose their errors accurately (Copur-Gencturk and Tolar, 2022; Daheim et al., 2024; Sonkar et al., 2024).

This perspective is increasingly prevalent in contemporary research on mathematics education using LLMs. In particular, there is a growing consensus that aligning LLMs to think like experienced educators rather than merely serving as answer-generating machines maximizes educational effectiveness. Recent studies have emphasized that, similar to human teachers, LLMs must engage in diagnosing errors and providing feedback based on students' solution processes when imparting Pedagogical Content Knowledge (PCK) (Jiang et al., 2024; Hu et al., 2025).

In this context, from a learning efficiency perspective, we emphasize the necessity for evaluation metrics that assess LLMs beyond simply providing direct answers. Specifically, such metrics should measure (1) whether LLMs can evaluate the similarity between student and teacher solutions with human-level precision, and (2) the accuracy with which LLMs can identify the initial error point in student solutions. To facilitate such eval-

[*] Equally contributed authors.

Findings of the Association for Computational Linguistics: EACL 2026, pages 1678–1704
March 24-29, 2026 ©2026 Association for Computational Linguistics

uation, it is essential to first establish evaluation datasets that include authentic solution processes generated by teachers. However, existing datasets, such as GSM8K, contain only mathematical problems accompanied by solved answers, lacking genuine teacher-generated solution processes (Cobbe et al., 2021). To address this gap, we augment the GSM8K dataset by incorporating real teacher-generated solutions, thus creating the Bi-GSM8K benchmark. Furthermore, we extend this benchmark to include student solution processes annotated explicitly with labels marking students' initial errors. Finally, we translate the augmented dataset into a bilingual Korean-English corpus, enabling the analysis of linguistic differences in mathematical problem-solving.

Using this benchmark, we evaluated how well various LLMs measure similarity between student and teacher solutions compared to human annotators. Our method achieved Pearson correlations of 0.89 and Spearman correlations of 0.88 on the English subset, and Pearson 0.89 and Spearman 0.87 on the Korean subset. The approach also demonstrated shorter computational latency than commercial APIs while maintaining comparable accuracy, indicating its practicality for real-time educational feedback. Additionally, leveraging annotations of students' initial error points, we assessed LLMs' ability to diagnose errors, with GPT-4o achieving approximately 95% accuracy and the best-performing open-source model attained 86% accuracy. The key contributions of this study are as follows:

- Construction and release of **Bi-GSM8K**, a bilingual Korean–English math education benchmark including teacher solutions, student solutions, and annotations of students' initial error locations.
- Proposal of a novel step-alignment metric for evaluating the logical similarity between student and teacher solutions in the mathematical problem-solving process.
- Evaluation of the accuracy in diagnosing initial error points in student solutions to verify the error analysis capabilities of LLMs.

2 Related Work

In this section, we review existing research related to mathematical education and LLMs by analyzing the characteristics influenced by problem-solving approaches, instructional methods, and linguistic differences.

2.1 Utilization and Limitations of LLMs in Mathematics Education

LLMs have shown promise in mathematics education, especially in problem-solving and tutoring. They achieve around 85.5% accuracy in algebraic problem-solving but lower performance in educational dialogue generation (Gupta et al., 2025). Stepwise error detection models improve the accuracy and pedagogical quality of LLM feedback (Daheim et al., 2024). Fine-tuning on datasets like MATHDIAL enhances feedback accuracy but challenges remain in error prevention, correction, and fully replacing human instruction (Macina et al., 2023). Limitations include difficulty handling novel errors, diagnosing student cognition, and filtering irrelevant content (Gupta et al., 2025; Daheim et al., 2024; Macina et al., 2023).

2.2 Challenges in Non-English Educational Environments

LLM performance on non-English math problems is still limited. GPT-4 achieved over 60% accuracy on the Chinese CMATH dataset (Wei et al., 2023), while other models performed worse. ChatGPT showed 66.7% accuracy on Korean secondary math problems (Nguyen et al., 2025), indicating some utility despite lower performance than English tasks. Multimodal assessments with datasets like KoNET reveal significant performance degradation in Korean high school settings (Park and Kim, 2025), underscoring persistent limitations in non-English contexts.

2.3 Research Trends of LLMs for Educational Purposes

Research on employing LLMs for educational purposes broadly divides into two streams. The first stream emphasizes enhancing mathematical problem-solving skills, utilizing LLMs to generate customized mathematical problems or solution methods to fine-tune student models and boost learning effectiveness (Liang et al., 2023). The second stream employs LLMs as tutors to simultaneously enhance students' learning outcomes and feedback generation capabilities (Scarlatos et al., 2025). Approaches include training LLMs using students' learning outcomes as reward signals, as well as employing schema-based strategies and role-based prompts to generate structured and pedagogically beneficial feedback (Dixit and Oates, 2024; Hu et al., 2025; Scarlatos et al., 2025).

These studies indicate the potential utility of LLMs in diverse pedagogical practices within mathematics education, such as problem generation, problem-solving, feedback provision, and instructional design. Nevertheless, concerns remain regarding the accuracy of generated outputs, the support for autonomous learning, and overall research reliability, underscoring ongoing areas for improvement.

In this context, recent studies such as ProcessBench (Zheng et al., 2025) and PRM800K (Lightman et al., 2023) have introduced process-level annotations to evaluate model-generated reasoning. However, these works primarily focus on validating the correctness of model-generated reasoning within monolingual settings. In contrast, our work constructs an English–Korean bilingual dataset and annotates step-level correspondences between teacher-authored and student-generated solutions, thereby providing an evaluation framework that extends beyond model-centric reasoning verification to enable fine-grained analysis of instructional feedback and cross-linguistic reasoning.

3 The Bi-GSM8K Dataset

What considerations are essential for LLMs to effectively teach mathematical problem-solving in a manner comparable to human instructors? A core prerequisite lies in the development of rigorous evaluation methodologies capable of assessing the degree to which LLM-generated responses emulate genuine teacher reasoning. In particular, these methodologies must facilitate the alignment of student-generated solutions with teacher-authored exemplars and enable the precise identification of the initial point of error.

To support learning and evaluation in intelligent tutoring systems, we introduce Bi-GSM8K, a bilingual dataset comprising elementary-level math problems, teacher solutions, and simulated student solutions. This dataset was initially constructed entirely in Korean by mathematics education experts and subsequently translated into English using GPT-4o, resulting in a high-quality bilingual resource.

Bi-GSM8K is built upon the existing English-based GSM8K dataset, which was systematically reconstructed to align with the Korean curriculum after automatic translation. The reconstruction process involved refining proper nouns, units, and vocabulary, as well as correcting unnatural expressions arising from translation. Mathematical expressions were standardized using the « » notation format consistent with GSM8K. The final dataset consists of 7,985 JSON-formatted entries, each containing curriculum-linked metadata, problem statements, teacher-authored correct solutions (correct_solution), intentionally flawed student solutions (error_solution), and annotations of students' initial error points.

```
{
  "area": "problem",
  "problem": "Byeongjin went fishing with his family
yesterday. Byeongjin caught 4 fish, his wife caught 1,
the eldest son caught 3, the younger son caught 2, and
the youngest daughter caught 5. Unfortunately, 3 of the
fish were too small and were released back. If each
fish yields 2 fillets, how many fillets can Byeongjin’s
family make?",
  "solution": "Four hats with 3 stripes each have a
total of 4×3=«43=12»12 stripes.\n Three hats with 4
stripes each have a total of 3×4=«34=12»12 stripes.\n
Six hats with no stripes have 6×0=«60=0»0 stripes.\n
And two hats with 5 stripes each have 2×5=«25=10»10
stripes.\n The total number of stripes on Byungjin’s
hats is 12+12+0+10=«12+12+0+10=34»34 stripes.\n ####
34",
  "correct_solution": {
    "step_1": "Byung-jin’s family caught 4 + 1 + 3 + 2
+ 5 = «4+1+3+2+5=15»15 fish.",
    "step_2": "15 - 3 = «15-3=12» I stored 12 fish.",
    "step_3": "Since you can obtain 2 fillets from each
fish, for 12 fish, you have 12 fish * 2 fillets per fish
= 24 fillets."
  },
  "error_solution": {
    "step_1": "4+1+3+2+5 = 15",
    "step_2": "15 * 2 = 30",
    "step_3": "Answer: 30 fillets"
  }
}
```

Figure 1: In the proposed Bi-GSM8K dataset, examples are expanded beyond the original GSM8K format, which included only “Problem” and “Solution” fields. Bi-GSM8K additionally provides the teacher solution (correct_solution) and an erroneous student solution (error_solution). Furthermore, the Bi-GSM8K dataset is offered as a bilingual Korean-English corpus.

In the dataset, both solution and correct_solution contain the correct solution; their content and meaning are identical, differing only in representation. The solution is provided in a string format, whereas the correct_solution is structured as a step-wise dictionary, enabling fine-grained analysis and facilitating alignment-based evaluation.

The correct_solution entries were authored by mathematics education and dataset construction experts and underwent rigorous review to ensure both computational accuracy and pedagogical validity. The error_solution entries were created by annotators with expertise in education rather than real student data, designed to incorporate realistic mathematical misconceptions reflecting typical student

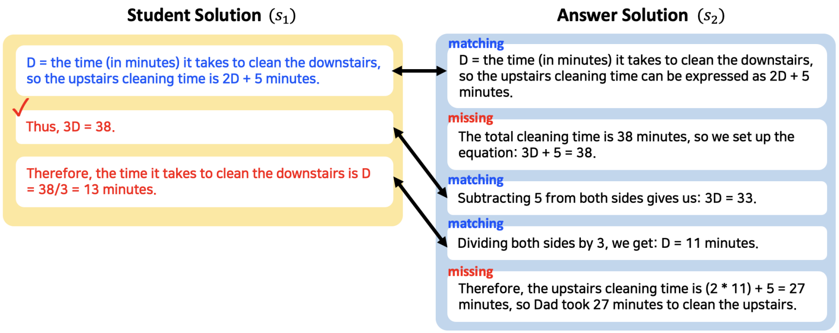
Student Solution (s_1)
D = the time (in minutes) it takes to clean the downstairs, so the upstairs cleaning time is 2D + 5 minutes.
✓
Thus, 3D = 38.
Therefore, the time it takes to clean the downstairs is D = 38/3 = 13 minutes.
Answer Solution (s_2)
matching
D = the time (in minutes) it takes to clean the downstairs, so the upstairs cleaning time can be expressed as 2D + 5 minutes.
missing
The total cleaning time is 38 minutes, so we set up the equation: 3D + 5 = 38.
matching
Subtracting 5 from both sides gives us: 3D = 33.
matching
Dividing both sides by 3, we get: D = 11 minutes.
missing
Therefore, the upstairs cleaning time is (2 * 11) + 5 = 27 minutes, so Dad took 27 minutes to clean the upstairs.

Figure 2: Upon submission of a student's solution to the system, an initial comparison and alignment with the answer solution is performed. This alignment employs Language Models (LMs), similarity functions, and the NW algorithm to systematically analyze omitted steps or extraneous information in the student's solution. Subsequently, an independent LLM-based error detection model operates separately from the alignment process to precisely identify the initial point of error within the student's solution.

errors. (In this paper, "student solutions" denote expert-crafted simulations of student errors, not real student answers.)

For initial error point annotation, annotators explicitly marked the first step at which a numerical or logical inconsistency occurred, regardless of consistency in subsequent steps. Minor expression variations, omissions, or stylistic changes were not considered errors. In cases with multiple errors, only the earliest logical error was annotated to align with the goal of initial error detection.

All annotations were cross-validated by mathematics education experts to ensure inter-annotator consistency and reliability. Detailed annotation guidelines, design principles for different error types, and the workflow for constructing erroneous student solutions are provided in the Appendix A. Representative examples are presented in Figure 1.

The Bi-GSM8K dataset offers the following key features:

- **Curriculum-Aligned Translation**: GSM8K was automatically translated into Korean and refined to match the national curriculum, with corrections to proper nouns, units, and translation errors. Correct solutions were thoroughly reviewed for accuracy.

- **Simulated Student Solution**: Incorporating consultation and review from mathematics education experts, we constructed student solution data that includes diverse error simulations designed to realistically emulate actual student responses.

- **Initial Error Annotation**: Each item identifies the first error in the student's solution, supporting fine-grained error diagnosis and targeted remediation by models.

4 Evaluation Metric

The Bi-GSM8K benchmark proposed in this study provides data enabling the evaluation of solution processes generated by LLMs by directly comparing them with those produced by experienced human teachers. To achieve this comparative evaluation, appropriate assessment methodologies are necessary. Specifically, this study introduces two similarity-based evaluation methods: (1) Ground Truth Alignment (GTA), which assesses how closely generated solutions align with correct teacher-generated solutions, and (2) Solution Error Detection (SED), which identifies the initial error point within student-generated solutions, thereby systematically evaluating the capabilities of language models.

4.1 Ground Truth Alignment

Evaluating narrative-style solutions of mathematical problems quantitatively is inherently challenging. Therefore, we propose GTA, which assesses solution quality based on the similarity between student solutions and ground-truth solutions. The similarity measurement involves three key steps: (1) Semantic similarity between solutions is measured using metrics such as cosine similarity of mean-pooled embeddings, Pearson correlation of mean-pooled embeddings, SemScore, and BERTScore (Salton et al., 1975; Pearson, 1895; Aynetdinov and Akbik, 2024; Zhang et al., 2019). (2) Based on the computed semantic similarity, the Needleman-Wunsch (NW) algorithm (Needleman and Wunsch, 1970) is applied to align the step sequences of teacher and student solutions. (3) Structural similarity is then evaluated based on the alignment results. The selection of similarity metrics utilized in this module, the approach to employing LLMs, and the specific manner in which LLMs are applied to compute each metric are comprehensively detailed in Appendix D.

Algorithm 1 NW Algorithm with Semantic Similarity for Ground Truth Alignment

```
1:  Input: $s1, s2, sim\_m, sim\_th$
2:  Output: $x\_aln, y\_aln$
3:  $m \leftarrow \texttt{len}(s1), n \leftarrow \texttt{len}(s2)$
4:  Initialize $bt\_table$ of size $(m+1, n+1)$
5:  for $i = 0$ to $m$ do
6:      $bt\_table[i][0] \leftarrow 1$                                   ▷ From up
7:  end for
8:  for $j = 0$ to $n$ do
9:      $bt\_table[0][j] \leftarrow 2$                                   ▷ From left
10: end for
11: for $i = 1$ to $m$ do
12:     for $j = 1$ to $n$ do
13:         $m\_sc \leftarrow score[i-1][j-1] + sim\_m[i-1][j-1]$
14:         $gap\_p \leftarrow gap\_u \times (1 - sim\_m[i-1][j-1])$
15:         $u\_sc \leftarrow score[i-1][j] - gap\_p$
16:         $l\_sc \leftarrow score[i][j-1] - gap\_p$
17:         $bt\_table[i][j] \leftarrow \text{argmax}(m\_sc, u\_sc, l\_sc)$
18:     end for
19: end for
20: $i \leftarrow m, j \leftarrow n$
21: Initialize $x\_aln, y\_aln$
22: while $i > 0$ or $j > 0$ do
23:     if $bt\_table[i][j] = 0$ then
24:         if $sim\_m[i-1][j-1] \geq sim\_th$ then
25:             Append aligned values to $x\_aln, y\_aln$
26:         end if
27:     end if
28:     Update indices $i$ and $j$
29: end while
30: Reverse $x\_aln, y\_aln$
31: Return $x\_aln, y\_aln$
```

In this study, we modified the conventional NW algorithm specifically for aligning mathematical solution processes. The NW algorithm numerically quantifies the similarity between two strings and identifies the most similar alignment, making it suitable for static similarity comparisons of lengthy texts. Typically, when comparing two texts, the NW algorithm assigns fixed penalty scores for character insertions or deletions. However, mathematical solution processes inherently involve both sequential order and detailed semantic content. Therefore, we adjusted gap penalties according to semantic similarity metrics, assigning smaller penalties to semantically similar sentences and larger penalties to dissimilar ones. Additionally, matching scores are computed based on substring similarities, enabling natural and precise alignment between solution steps. This allows clear identification of differences between two solutions and facilitates alignment consistent with the problem-solving flow.

Algorithm 1 describes the proposed procedure for computing alignment scores between two solutions (examples in Figure 2). Here, s_1 represents the teacher-generated solution with m steps, and s_2 denotes the student-generated solution with n steps. The inputs are two sequences (s_1, s_2), a similarity matrix (sim_m) indicating semantic similarities between solution steps, and a similarity threshold (sim_th). A backtracking table bt_table for dynamic programming is initialized (line 4) with dimensions $(m+1) \times (n+1)$, storing directional moves for reconstructing optimal alignments. The first row and column of bt_table are initialized to represent leftward and upward movements, respectively.

Next, a backtracking table bt_table for dynamic programming is initialized (line 4). The table has dimensions $(m+1) \times (n+1)$, with each cell recording the optimal move direction for backtracking. The first row and first column of bt_table are initialized with left (represented by 2) and upward (represented by 1) movements, respectively.

Subsequently, the remainder of bt_table is filled using two nested loops. At each cell (i, j), values for three possible movements are calculated, and the maximum value is selected and recorded in $bt\_table[i][j]$. Diagonal movements (matches) add the value from the diagonal cell and $sim\_m[i-1][j-1]$; upward movements (deletions) add penalties to values from the cell above; leftward movements (insertions) add penalties to values from the cell to the left. Once the table is fully populated, an optimal alignment between the sequences is determined by backtracking through bt_table.

During the backtracking phase, alignments are determined according to each cell's recorded movement direction. For diagonal movements (represented by 0), if the similarity is below the threshold sim_th, the element $s_1[i-1]$ is aligned as an "omission". If the similarity meets or exceeds sim_th, the algorithm checks for duplicate alignments. Specifically, if $s_2[j-1]$ is already aligned with another element in y_aln, the algorithm compares similarity scores between the existing and current alignments. If the existing alignment has a higher similarity score, the current element $s_1[i-1]$ is aligned as an "omission"; otherwise, the existing alignment is replaced with the current one. If no duplication occurs, the two elements are directly aligned. For upward movements (represented by 1), the element $s_1[i-1]$ is aligned as an "omission". For leftward movements (represented by 2), an "unnecessary" is aligned with the element $s_2[j-1]$. After backtracking completes, the aligned sequences $x\_aln.reverse()$ and $y\_aln.reverse()$, along with the similarity matrix sim_m, are returned in the correct order.

Based on the alignment results, a score of 1 is assigned to each matched step between the teacher and student solutions, while unmatched or extraneous steps are assigned a score of 0. The final similarity score is computed by summing the individual step scores and dividing by the total number of steps, yielding a value between 0 and 1.

The proposed method, GTA, evaluates logical consistency by precisely aligning reasoning steps, thereby enabling fine-grained diagnosis of student errors and facilitating personalized feedback. The resulting similarity scores can be used to track learner understanding, analyze problem difficulty, improve automated grading, and enhance adaptive learning, supporting personalized education.

4.2 Solution Error Detection

Identifying the initial error step is particularly crucial in personalized tutoring scenarios. Precisely pinpointing the moment when a student deviates from the correct problem-solving strategy enables the system to gain deeper insights into the student's conceptual understanding. This module aims to accurately identify the initial erroneous step in a student's mathematical problem-solving process.

You are given a math problem, a correct answer solution, and a student solution. Your task is to compare the correct solution to the student solution and output the step number where the student's error begins. If the student's solution is completely correct, output 0.
Problem: Seongjin is stranded on a deserted island. He needs salt to season fish. He collected 2 liters of seawater in an old bucket. If the water contains 20% salt, how many milliliters of salt will Seongjin get when all the water evaporates?
Answer Solution:
{
"step_1":"First, find out how many liters of the seawater is salt: 2L * 20% = «2*0.2=0.4»0.4L",
"step_2":"Then, multiply this amount by 1000 ml/L to find how much salt Seongjin gets: 0.4L * 1000ml/L = «0.4*1000=400»400ml"
}
Student Solution:
{
"step_1":"220 = «220=40»40ml of salt is obtained."
}
Q: Is the Student Solution incorrect? Write only the step number with the first error or 0 if no error is found.
A: 1
...
Problem: `{problem}`
Answer Solution: `{answer_solution}`
Student Solution: `{student_solution}`
Q: Is the Student Solution incorrect? Write only the step number with the first error or 0 if no error is found.
A:

Figure 3: Prompt used for initial error detection in Bi-GSM8K.

The initial error detection module proposed in this study employs a 3-shot learning approach utilizing open-source LLMs, enabling accurate error identification with only a small number of examples. Figure 3 illustrates a partial structure of the prompt, which presents three exemplars. Each exemplar consists of a mathematical problem, a correct solution (Answer Solution), a student solution (Student Solution), and a question (Q) asking for the step at which the student's first error occurs. The model compares the teacher's solution with the student's solution to predict the step number of the initial error, returning 0 if no error is detected.

Due to space constraints, the complete prompts and the rationale behind exemplar selection are provided in Appendix I. To further analyze the impact of including the teacher's solution on model performance, experiments were also conducted using prompt variants that exclude the teacher's solution, with detailed results presented in Appendix F. Additionally, preliminary experimental results and analyses of a simple integration setup in which GTA outputs are incorporated into SED are reported in Appendix E, providing evidence supporting the validity of the design choice to develop GTA and SED as independently designed and operated tasks.

5 Experiment

5.1 Experiment Settings and Models

In this study, we evaluated various open-source LLMs using the mathematical Bi-GSM8K dataset. The models selected for experimentation were assessed comprehensively based on multilingual processing capabilities, mathematical reasoning abilities, response consistency, and computational efficiency. Models ranging from 7B to 8B parameters were specifically chosen due to their balance between performance and practicality, making them suitable for real-world educational contexts. In addition, the GTA module employs various semantic similarity functions to perform step-level alignment between student solutions and reference answers. These functions are used to identify semantically similar step pairs, facilitating the alignment process based on their degree of semantic closeness. Among them, the BERTScore experiments leverage multiple pretrained transformer models to enable fine-grained semantic analysis. The primary hyperparameters for the open-source models were set to a temperature of 1, a top_p of 0.75, a top_k of 40, and num_beams of 4. For GPT-4o, the temperature was adjusted to 0.75 to enhance response consistency and stability.

Similarity	Model	Threshold	Pearson		Spearman		Latency	Peak Memory
			EN	KO	EN	KO	(sec)	(MB)
-	GPT-4o (Hurst et al., 2024)	–	0.8645	0.9168	0.8605	**0.9394**	3.372	–
	GPT-4o-mini (OpenAI, 2024a)	–	0.8887	**0.9249**	0.8712	0.9382	3.614	–
	Claude-Sonnet-4 (Anthropic, 2024)	–	0.8975	0.9009	0.8955	0.935	4.026	–
	Gemini-2.5-Flash (Comanici et al., 2025)	–	**0.9010**	0.9197	**0.8978**	0.9375	13.118	–
Cosine Pearson Semscore	Llama-3.1-8B-Instruct (Dubey et al., 2024)	0.5	0.8823	0.8795	0.8618	0.8685	0.427	28681.48
	DeepSeek-R1-Distill-Llama-8B (Guo et al., 2025)	0.5	0.8873	0.8771	0.8706	0.8651	0.429	28681.48
	DeepSeek-llama3.1-Bllossom-8B (UNIVA-Bllossom, 2025)	0.5	0.8895	0.8736	0.8659	0.8613	0.389	28658.35
	Qwen2.5-7B-Instruct (Hui et al., 2024)	0.5	0.8901	0.8769	0.8628	0.8634	0.406	27178.01
	DeepSeek-R1-Distill-Qwen-7B (Guo et al., 2025)	0.65	0.8893	0.7798	0.8657	0.7788	0.373	27178.21
	DeepSeek-R1-Distill-Qwen-7B-Multilingual (LightBlue, 2025)	0.5	0.8901	0.7899	0.8628	0.7838	0.398	27162.12
	Mistral-7B-Instruct-v0.3 (Jiang et al., 2023)	0.5	0.8816	0.8884	0.8670	0.8631	0.501	27221.69
	Phi-4-mini-instruct (Abouelenin et al., 2025)	0.5	0.8901	0.8896	0.8628	0.8629	0.183	14728.41
BertScore	bart-large (Lewis et al., 2019)	0.5	0.8903	0.8898	0.8698	0.8630	**0.070**	**703.91**
	bert-large-uncased (Devlin et al., 2019)	0.5	**0.8942**	**0.8929**	**0.8752**	0.8665	0.113	1010.21
	deberta-v2-xlarge-mnli (He et al., 2020)	0.65	0.8881	0.8912	0.8668	**0.8747**	0.213	2683.69
	roberta-large (Liu et al., 2019)	0.5	0.8901	0.8901	0.8628	0.8628	0.109	1040.56

Table 1: Correlation between model-generated similarity scores and human-evaluated similarity scores (Pearson, Spearman) in English (EN) and Korean (KO), along with latency and memory usage for each model. The highest value in each category is highlighted in **black bold**.

5.2 Ground Truth Alignment

This section presents an analysis of similarity evaluation performance between student and teacher solutions using the Bi-GSM8K dataset. Table 1 summarizes the performance of the GTA module across various similarity metrics, model configurations, and threshold settings on both English and Korean datasets, alongside computational efficiency metrics such as latency and peak memory usage. Performance was evaluated by comparing the similarity scores predicted by the models against those provided by five human annotators using Pearson (Pearson, 1895) and Spearman (Spearman, 1961) correlation coefficients. Detailed evaluation guidelines and inter-annotator agreement analyses are provided in Appendix G and Appendix H, respectively. All threshold and efficiency metrics are reported based on the average values over both language datasets.

The evaluation was conducted on 500 randomly selected problems from Bi-GSM8K. Considering the sensitivity of similarity-based evaluation to threshold settings, experiments were performed over five predefined thresholds within the range [0.5, 0.6, 0.7, 0.8, 0.9]. Table 1 reports only the results corresponding to the optimal threshold for each model. All evaluations were carried out using the same test set and consistent hyperparameter settings to prevent overfitting. Examples of alignment results are included in Appendix J.

Among commercial LLM APIs, Gemini-2.5-Flash achieved the highest Pearson correlation on the English subset (0.9010), while GPT-4o-mini led on the Korean subset (0.9249). Regarding Spearman correlation, Gemini-2.5-Flash again ranked highest on English (0.8978), and GPT-4o achieved the top score on Korean (0.9394). Within the GTA module variants, BERTScore-based models exhibited superior performance and efficiency. Specifically, `bert-large-uncased` attained Pearson correlations of 0.8942 (EN) and 0.8929 (KO), along with a Spearman correlation of 0.8752 (EN). Meanwhile, `deberta-v2-xlarge-mnli` recorded the highest Spearman correlation on Korean (0.8747). These open-source configurations achieved performance within approximately 0.03 points of commercial models on average, indicating comparable alignment quality in practice. This underscores the capacity of context-aware embeddings to capture semantic similarity at a human-like level, enabling precise step-level alignment between student and teacher solutions. Furthermore, these results highlight the importance of semantic fidelity in alignment tasks and demonstrate the practical potential of open-source models.

In terms of computational efficiency, large-scale LLMs exhibited average inference latencies of around 0.4 seconds per problem, whereas BERT-based models offered substantially faster runtimes, e.g., `bart-large` and `bert-large-uncased` required only 0.07 and 0.11 seconds, respectively. Notably, the commercial small-scale model GPT-4o-mini incurred a latency of 3.614 seconds, significantly slower than the open-source counterparts. Similar trends were observed for memory consumption: large LLMs demanded over 27,000 MB, while BERT-based models generally operated under 3,000 MB, with `bart-large` requiring approximately 700 MB, demonstrating feasibility in resource-constrained environments.

Model	Accuracy		Throughput	Latency	Peak Memory
	EN	KO	(requests/sec)	(sec)	(MB)
GPT-4o	**94.4**	**95.8**	0.1309	8.39	-
Claude-Sonnet-4	87.8	90.2	0.1885	5.31	-
Gemini-2.5-Flash	90.4	91.4	**0.203**	**4.91**	-
Llama-3.1-8B-Instruct	80.4	75.2	0.0014	711.60	10896.85
DeepSeek-R1-Distill-Llama-8B	55.4	57.8	0.0014	720.06	10898.43
DeepSeek-llama3.1-Bllossom-8B	63.2	62.0	0.0014	716.86	10851.40
Qwen2.5-7B-Instruct	79.4	**86.0**	0.0013	783.91	11279.73
DeepSeek-R1-Distill-Qwen-7B	82.8	80.4	0.0013	801.63	11328.93
DeepSeek-R1-Distill-Qwen-7B-Multilingual	**86.4**	83.4	0.0012	847.68	11281.79
Mistral-7B-Instruct-v0.3	67.0	67.0	0.0009	1168.91	15120.72
Phi-4-mini-instruct	76.6	78.6	**0.0029**	**355.01**	**8075.12**

Table 2: Comparative evaluation of Solution Error Detection scores, thresholds, and memory usage for each model

5.3 Solution Error Detection

In this section, we evaluates the accuracy of various LLMs in detecting students' initial errors during the process of solving mathematical problems. Table 2 summarizes the performance of multiple LLMs on the SED task, reporting accuracy on both English (EN) and Korean (KO) datasets, along with computational efficiency metrics such as throughput (requests per second), latency (seconds), and memory usage (MB). The evaluation was conducted on 500 randomly selected items from the Bi-GSM8K dataset. Ground truth for the initial error points was provided by mathematics and data construction experts, allowing accuracy to be computed without additional human annotation. Detailed examples of initial error detection are presented in Appendix K.

Among commercial LLMs APIs, GPT-4o achieved the highest accuracy in our experiments, recording 94.4% on English and 95.8% on Korean. In terms of computational efficiency, Gemini-2.5-Flash demonstrated the best performance, processing 0.2037 requests per second with an average latency of 8.39 seconds.

Among open-source multilingual LLMs, models based on the Qwen architecture generally outperformed single-language or simpler models. For instance, DeepSeek-R1-Distill-Qwen-7B-Multilingual achieved accuracies of 86.4% in English and 83.4% in Korean. In contrast, the single-language model Qwen2.5-7B-Instruct improved English accuracy to 79.4% but saw a decline in Korean accuracy from 86.0% to 83.4%. These results suggest that multilingual training contributed to improvements in English performance but potentially hindered Korean performance. This may be due to the dominance of English-centric representational structures and semantic frameworks learned during multilingual training, which could impair representation and reasoning capabilities for structurally distinct languages such as Korean. In other words, multilingual training does not guarantee uniform performance gains across languages; dominant languages can bias the model, adversely affecting others.

Conversely, lightweight models based on the Llama series showed relatively lower accuracy. Llama-3.1-8B-Instruct achieved 80.4% accuracy on English and 75.2% on Korean, while DeepSeek-R1-Distill-Llama-8B scored 55.4% and 57.8%, respectively. Notably, DeepSeek-llama3.1-Blossom-8B, trained to think in English and output in the input language, exhibited decreased performance when Korean data was included. This indicates that the English-centric reasoning strategy was insufficiently adapted to Korean contexts or that the semantic quality of the training data was inadequate for mathematical reasoning and error detection.

Regarding computational efficiency, the Phi-4-mini-instruct model showed high resource efficiency with throughput of 0.0029 requests per second, latency of 355.01 seconds, and memory usage of 8075.12 MB. However, its low accuracy renders it unsuitable for tutoring scenarios requiring real-time feedback. Most open-source models demonstrated higher throughput and lower latency compared to GPT-4o, providing better real-time responsiveness, but they still exhibited limitations in solving complex problems and in-depth language understanding.

For a more intuitive understanding of the GTA and SED experimental results presented in this paper, various visualizations are included in Appendix L.

6 Conclusion

This study introduces Bi-GSM8K, a bilingual English-Korean benchmark dataset for mathematical problem solving. Constructed from detailed solution processes of both students and teachers, Bi-GSM8K serves as a foundational resource for math education and enables comprehensive evaluation of similarity between student and teacher solutions. GTA evaluation demonstrates that several competitive open-source models achieve high agreement with human raters while offering greater computational efficiency compared to commercial models. Notably, the combination of BERTScore with `bert-large-uncased` and the NW algorithm attains performance comparable to commercial systems. In the SED task, open-source models perform slightly below commercial counterparts, with an accuracy gap of approximately 10 percentage points, indicating practical applicability. Language-specific analyses further reveal performance differences between English and Korean, underscoring the influence of linguistic properties and training data composition. In addition, experiments conducted separately on the English and Korean subsets reveal distinct performance disparities by language, suggesting that alignment and error detection capabilities vary according to linguistic characteristics and training data distribution.

Limitations

Dataset: Elementary-Level Scope, Simulated Student Solutions, and Language Selection Considerations

The dataset focuses on elementary-level mathematics problems, thereby limiting its applicability to specific grade levels and problem difficulties. The choice to concentrate on elementary mathematics was motivated by its suitability as an initial benchmark for evaluating the capability of LLMs to accurately track students' cognitive processes and detect errors in a stepwise manner. Elementary math problems generally feature clearer and more structured solution paths, which facilitate systematic analysis and experimental control in early-stage investigations. In contrast, middle- and high-school level problems require more complex reasoning, rendering them less suitable for preliminary studies. Future work will extend this research to encompass higher-level problems that demand more sophisticated reasoning.

Moreover, the student-generated solutions in this dataset were simulated by domain experts under assumptions of typical student behavior. Consequently, they may not fully capture the diversity of cognitive processes and error patterns exhibited by real students. Nonetheless, these simulations were carefully constructed and validated by mathematics education experts to reflect common error types frequently observed among students. Collecting authentic student solutions presents challenges due to ethical considerations and privacy constraints; therefore, future research should aim to develop datasets encompassing a broader range of problem types, languages, and genuine student solutions.

The dataset is bilingual, comprising both Korean and English. Korean, being the native language of the authors, facilitated efficient data construction and validation and, as a relatively low-resource language in the context of LLM training, provides a suitable testbed for examining the generalization capabilities of multilingual models and for analyzing language-specific error patterns. English, as the primary language used for training most LLMs, serves as a critical benchmark for model evaluation. The bilingual nature of the dataset enables analyses of semantic alignment, variation in expression, and cross-linguistic bias, and, by leveraging the typological differences between Korean and English, provides a meaningful framework for evaluating models' cross-linguistic understanding capabilities.

Evaluation in Real Educational Environments

While this research evaluates system performance through quantitative data-driven analyses, direct verification of its applicability in actual classroom or tutoring environments has not been performed. In real educational settings, factors such as student responses, class dynamics, and teacher interventions significantly influence system effectiveness. Therefore, comprehensive evaluations of system practicality and educational efficacy require implementation in authentic educational contexts. Future studies should conduct experiments involving teachers and students to measure educational effectiveness and derive feedback mechanisms and interface improvements that meet real-world demands.

Limitations and Improvement Directions for Solution Alignment Representation

In this study, we performed sentence-level alignment between student-generated and correct (teacher-generated) solutions. However, this approach exhibits limitations in alignment accuracy, as a single sentence may often encompass multiple solution steps. To precisely model students' reasoning processes and effectively capture the intricate relationships between student and teacher solutions, a hierarchical and multi-layered representational approach is required. Consequently, developing novel alignment methods capable of reflecting such complex structures is proposed as an important direction for future research.

Acknowledgments

This work was supported by Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (RS-2023-00216011, Development of artificial complex intelligence for conceptually understanding and inferring like human).

References

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

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Anthropic. 2024. Introducing claude4.

Ansar Aynetdinov and Alan Akbik. 2024. Semscore: Automated evaluation of instruction-tuned llms based on semantic textual similarity. *arXiv preprint arXiv:2401.17072*.

Deborah Loewenberg Ball, Mark Hoover Thames, and Geoffrey Phelps. 2008. Content knowledge for teaching: What makes it special?

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Yasemin Copur-Gencturk and Tammy Tolar. 2022. Mathematics teaching expertise: A study of the dimensionality of content knowledge, pedagogical content knowledge, and content-specific noticing skills. *Teaching and Teacher Education*, 114:103696.

Nico Daheim, Jakub Macina, Manu Kapur, Iryna Gurevych, and Mrinmaya Sachan. 2024. Stepwise verification and remediation of student reasoning errors with large language model tutors. *arXiv preprint arXiv:2407.09136*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186.

Prakhar Dixit and Tim Oates. 2024. Sbi-rag: Enhancing math word problem solving for students through schema-based instruction and retrieval-augmented generation. *arXiv preprint arXiv:2410.13293*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Adit Gupta, Jennifer Reddig, Tommaso Calo, Daniel Weitekamp, and Christopher J MacLellan. 2025. Beyond final answers: Evaluating large language models for math tutoring. *arXiv preprint arXiv:2503.16460*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Bihao Hu, Jiayi Zhu, Yiying Pei, and Xiaoqing Gu. 2025. Exploring the potential of llm to enhance teaching plans through teaching simulation. *npj Science of Learning*, 10(1):7.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. 2024. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Zhuoxuan Jiang, Haoyuan Peng, Shanshan Feng, Fan Li, and Dongsheng Li. 2024. Llms can find mathematical reasoning mistakes by pedagogical chain-of-thought. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pages 3439–3447. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Zhenwen Liang, Wenhao Yu, Tanmay Rajpurohit, Peter Clark, Xiangliang Zhang, and Ashwin Kaylan. 2023. Let gpt be a math tutor: Teaching math word problem solvers with customized exercise generation. *arXiv preprint arXiv:2305.14386*.

LightBlue. 2025. Deepseek-r1-distill-qwen-7b- multilingual.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Jakub Macina, Nico Daheim, Sankalan Pal Chowdhury, Tanmay Sinha, Manu Kapur, Iryna Gurevych, and Mrinmaya Sachan. 2023. Mathdial: A dialogue tutoring dataset with rich pedagogical properties grounded in math reasoning problems. *arXiv preprint arXiv:2305.14536*.

Mary L McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica*, 22(3):276–282.

Saul B Needleman and Christian D Wunsch. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453.

Phuong-Nam Nguyen, Quang Nguyen-The, An Vu-Minh, Diep-Anh Nguyen, and Xuan-Lam Pham. 2025. On the robustness of chatgpt in teaching korean mathematics. *arXiv preprint arXiv:2502.11915*.

OpenAI. 2024a. Gpt-4o mini: Advancing cost-efficient intelligence.

OpenAI. 2024b. Learning to reason with large language models. `https://openai.com/index/learning-to-reason-with-llms/`.

Sanghee Park and Geewook Kim. 2025. Evaluating multimodal generative ai with korean educational standards. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 671–688.

Karl Pearson. 1895. Vii. note on regression and inheritance in the case of two parents. *proceedings of the royal society of London*, 58(347-352):240–242.

Gerard Salton, Anita Wong, and Chung-Shu Yang. 1975. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.

Alexander Scarlatos, Naiming Liu, Jaewook Lee, Richard Baraniuk, and Andrew Lan. 2025. Training llm-based tutors to improve student learning outcomes in dialogues. *arXiv preprint arXiv:2503.06424*.

Shashank Sonkar, Kangqi Ni, Sapana Chaudhary, and Richard G Baraniuk. 2024. Pedagogical alignment of large language models. *arXiv preprint arXiv:2402.05000*.

Charles Spearman. 1961. The proof and measurement of association between two things.

Those Who Understand. 1986. Knowledge growth in teaching. *Educational Researcher*, 15(2):4–14.

UNIVA-Bllossom. 2025. Deepseek-llama3.1-bllossom-8b.

Tianwen Wei, Jian Luan, Wei Liu, Shuang Dong, and Bin Wang. 2023. Cmath: Can your language model pass chinese elementary school math test? *arXiv preprint arXiv:2306.16636*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. Processbench: Identifying process errors in mathematical reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1009–1024.

Qihuang Zhong, Kang Wang, Ziyang Xu, Juhua Liu, Liang Ding, and Bo Du. 2024. Achieving> 97% on gsm8k: Deeply understanding the problems makes llms better solvers for math word problems. *arXiv preprint arXiv:2404.14963*.

A Detailed Annotation Process for the Bi-GSM8K Dataset

In this appendix, we provide a detailed description of the annotation process for the Bi-GSM8K dataset. The process was conducted under the overall supervision of an expert holding a Ph.D. in mathematics education, who was responsible for providing methodological guidance, establishing annotation criteria, and performing final validation. Initial dataset construction—including the creation of problem instances and draft student solutions—was carried out by dataset construction experts in accordance with predefined annotation guidelines. All annotation outputs were subsequently reviewed and refined by the supervising expert to ensure pedagogical validity and overall consistency.

Prior to dataset construction, detailed annotation guidelines were established. In particular, the error_solution entries were designed to avoid arbitrary or unrealistic mistakes. When generating erroneous student solutions, annotators incorporated a range of learner error patterns commonly observed in real educational settings. These include errors occurring at different steps of the solution process, cases where students skip intermediate steps and proceed several steps ahead after partial instruction, attempts to continue problem solving despite insufficient conceptual understanding, and situations requiring repeated explanations. Scenarios in which students are unable to initiate a solution at all were also explicitly included as a distinct error type.

When authoring student solution processes, annotators intentionally injected mathematical or logical errors while maintaining an overall coherent solution trajectory. To realistically simulate learner behavior, student solutions were constrained not to deviate entirely from the structure of the correct solution, thereby reflecting errors arising from partial understanding rather than random reasoning. All mathematical expressions within solution steps were consistently formatted using the « » notation to maintain compatibility with GSM8K.

The initial error point was defined as the earliest step at which the learner fails to resolve a numerical or logical inconsistency, causing the problem-solving process to break down for the first time. Even if subsequent steps appeared internally consistent, they were not considered when identifying the initial error point. Minor variations in expression, trivial omissions, or stylistic differences that did not affect logical validity were not treated as errors.

In cases where multiple errors occurred within a single student solution, only the earliest logically invalid step was annotated, in accordance with the objective of initial error detection. All annotation results were cross-validated by mathematics education experts to ensure inter-annotator consistency and overall annotation reliability.

B Analysis of Problem Lengths

In this appendix, we present a quantitative statistical analysis of problem lengths in the Bi-GSM8K dataset utilized in our experiments. The dataset comprises mathematical problems presented in both Korean and English, each containing 500 items. A statistical summary of the problem lengths is provided in Table 3.

Metric	English	Korean
Count	500.000	500.000
Mean	237.004	122.876
Std	90.912	44.501
Min	62.000	36.000
25%	173.000	90.000
50%	220.000	117.000
75%	281.000	145.250
Max	592.000	304.000

Table 3: Statistical Summary of Problem Lengths

The average length of English items was approximately 237 characters, significantly longer than the average length of Korean items (122.88 characters). This discrepancy likely arises from the automatic translation of Korean data into English, resulting in generally more detailed explanations or structurally longer sentences in the English items. Additionally, the standard deviation of English problems (90.91) is more than double that of Korean problems (44.50), indicating a broader and more varied length distribution in the English dataset.

Regarding minimum lengths, English problems contain at least 62 characters, while Korean problems have a minimum of 36 characters, suggesting that English texts generally include more information. The maximum length further highlights this difference, with English problems reaching up to 592 characters compared to the 304 characters of Korean problems.

Quartile-based metrics exhibit similar trends, consistently showing higher values for English problems compared to their Korean counterparts, thereby reinforcing the structural length disparity across the entire dataset range. Notably, the median length of English problems is 220 characters, approximately 1.88 times greater than the Korean median of 117 characters.

These results might reflect the explicit and detailed sentence structures often required by the English language during translation, as well as potential adjustments made by generative models to accommodate stylistic differences between languages. Such findings underscore the necessity of considering language-specific perceptions of difficulty and interpretative approaches in subsequent analyses.

C Analysis of Solution Lengths

In this appendix, we present a quantitative statistical analysis of the lengths and steps of student-generated and correct solutions from the Bi-GSM8K dataset used in our experiments. The dataset comprises problems provided in two languages, Korean and English, each consisting of 500 items.

C.1 Analysis of Solution Length by Steps

This section analyzes the lengths of complete solutions written in Korean and English for both correct and student-generated solutions within the dataset. The first analysis considers each solution as an integrated unit without dividing it into individual steps.

Metric	Correct Solution	Student Solution
Count	500.000	500.000
Mean	278.596	181.296
Std	136.449	117.530
Min	36.000	8.000
25%	174.750	99.750
50%	249.500	169.000
75%	353.000	244.000
Max	1006.000	689.000

Table 4: Statistical Summary of Solution Lengths (English)

The statistical analysis results for English solutions are summarized in Table 4. The average length of correct (teacher-generated) solutions is approximately 278.60 characters, roughly 1.54 times longer than student-generated solutions, which average 181.30 characters. Notably, the standard deviation for correct solutions (136.45) is considerably higher compared to that for student solutions (117.53), indicating that correct solutions not only tend to be lengthier but also exhibit greater variability in structural complexity and explanatory depth.

Metric	Correct Solution	Student Solution
Count	500.000	500.000
Mean	185.456	122.008
Std	82.344	75.418
Min	46.000	5.000
25%	123.000	70.000
50%	168.000	112.500
75%	237.000	166.000
Max	631.000	389.000

Table 5: Statistical Summary of Solution Lengths (Korean)

The statistical results for Korean solutions are summarized in Table 5, displaying trends similar to those observed in the English solutions. The average length of correct (teacher-generated) solutions in Korean is approximately 185.46 characters, approximately 1.5 times longer than student-generated solutions, which average 122.01 characters. The standard deviation of correct solutions (82.34) is slightly greater than that of student solutions (75.42), suggesting more variability due to detailed explanatory content. Furthermore, the maximum length of correct solutions (631 characters) substantially surpasses that of student solutions (389 characters).

Overall, correct solutions consistently exhibit greater length compared to student solutions, reflecting their more detailed and stepwise explanatory nature. This pattern is consistent across both languages, reinforcing the observation that teacher-generated solutions generally present higher complexity and explanatory completeness.

The subsequent analysis separately examines solution lengths at the individual step level.

Metric	Correct Solution	Student Solution
Count	1724	1340
Mean	80.80	67.65
Std	34.27	32.12
Min	6	6
25%	57	48
50%	74	63
75%	99	83
Max	265	282

Table 6: Statistical Summary of Step Lengths (English)

Tables 6 and 7 provide statistical summaries of step-by-step solution lengths in English and Korean, respectively.

In English, the average step length of correct solutions (80.80 characters) exceeds that of student solutions (67.65 characters). Additionally, the

Metric	Correct Solution	Student Solution
Count	1724	1340
Mean	53.79	45.53
Std	17.22	17.57
Min	7	5
25%	43	36
50%	51	44
75%	64	55
Max	135	157

Table 7: Statistical Summary of Step Lengths (Korean)

standard deviation and maximum-minimum values show a broader distribution for correct solutions, indicating that these solutions may contain more detailed and complex explanations.

Similar trends appear in Korean solutions. Correct solutions have an average step length of 53.79 characters, longer than the 45.53 characters for student solutions. Standard deviations for both groups were comparable, while maximum lengths were higher in student solutions, indicating the existence of some student solutions with extensive explanations.

Generally, correct solutions have longer and more detailed step explanations than student solutions, although exceptions exist regarding maximum-minimum lengths and range distributions.

Step	Correct Solution	Student Solution
1	82.47	64.92
2	79.81	66.95
3	79.90	71.71
4	81.15	70.64
5	83.47	70.63
6	74.74	74.47
7	73.58	71.00
8	69.75	-

Table 8: Average Step Length per Step Number (English)

Step	Correct Solution	Student Solution
1	54.20	43.04
2	53.32	45.39
3	53.84	48.37
4	54.81	48.73
5	53.26	47.54
6	50.98	49.94
7	51.50	54.00
8	53.75	-

Table 9: Average Step Length per Step Number (Korean)

Tables 8 and 9 display the average solution length by individual steps for English and Korean, respectively.

For English solutions, the average length of correct solutions consistently surpasses that of student solutions across the initial five steps, with the largest discrepancy observed in step 1 (82.47 vs. 64.92 characters). Differences decrease in subsequent steps, with steps 6–7 showing minimal divergence, and step 8 lacking student solution data, suggesting that correct solutions tend to provide longer, more detailed explanations early in the solution process.

Korean solutions reveal a similar pattern, with correct solutions typically longer than student solutions across most steps, though student solutions exceed correct solutions at step 7. Similar to English data, step 8 lacks student-generated solution data. Compared to English, differences in length per step are generally smaller in Korean solutions.

Overall, in both languages, the difference in length per step decreases as solutions progress, with correct solutions consistently providing more comprehensive explanations.

C.2 Analysis of Solution Steps

This section analyzes the number of steps in the Correct Solutions and Student Solutions within the dataset. The number of solution steps is a crucial metric indicating how granularly the solution process is articulated, serving as an essential factor for evaluating the detail and complexity of the solutions.

Metric	Correct Solution	Student Solution
Count	500	500
Mean	3.448	2.680
Std	1.349	1.309
Min	2	1
25%	2	2
50%	3	3
75%	4	3
Max	8	7

Table 10: Statistical Summary of Step Counts by Solution Type

Table 10 summarizes statistical measures for the step counts of both groups. The analysis was conducted on 500 solution samples from each group.

The Correct Solutions exhibited an average of 3.448 steps, noticeably higher than the Student Solutions, which averaged 2.680 steps. This indicates

Metric	Input Unit	Embedding Method	Similarity Computation
Mean-pooled Embedding Similarity	Sentence-level vector	Mean pooling over all tokens	Cosine similarity between vectors
Mean-pooled Embedding Correlation	Sentence-level vector	Mean pooling over all tokens	Pearson correlation per dimension
SemScore	[CLS] token embedding	Single [CLS] token embedding	Cosine similarity between [CLS] embeddings
BERTScore	Token-level	All token embeddings	Alignment optimization-based F1 score

Table 11: Summary of semantic similarity metrics used for comparing student and teacher solutions.

a tendency for Correct Solutions to provide more detailed and finely segmented explanations. The standard deviation of step counts was similarly around 1.3 for both groups, suggesting comparable variability in the number of solution steps.

Regarding the minimum number of steps, Student Solutions included instances starting from a single step, whereas Correct Solutions always began from at least two steps. Additionally, the 75th percentile step count was 4 for Correct Solutions and 3 for Student Solutions, reinforcing the observation that Correct Solutions generally involve more steps.

For maximum step counts, Correct Solutions extended up to 8 steps, whereas Student Solutions reached a maximum of 7 steps, indicating slightly less granularity in student-generated explanations.

These results suggest that Student Solutions tend to provide briefer explanations or omit certain steps compared to Correct Solutions. Thus, step count analysis serves as a valuable measure for assessing the completeness and structural detail of student-generated solutions.

D Semantic Similarity Metrics Used in GTA

This appendix provides a detailed description of the four representative semantic similarity metrics employed in our GTA evaluation. These metrics were selected to capture not only surface-level textual overlap but also deeper semantic correspondences at both sentence and token levels. Table 11 summarizes the input granularity, embedding strategy, and similarity computation method for each metric.

Mean-pooled Embedding Cosine Similarity For this metric, we extract token embeddings from a Transformer-based language model and compute a sentence-level embedding by performing mean pooling across all token embeddings. Cosine similarity is then computed between the sentence embeddings of the student and teacher solutions. This is a standard approach that aggregates semantic information across the sentence and allows for holistic comparison of overall meaning.

Mean-pooled Embedding Pearson Correlation This method calculates the Pearson correlation coefficient across each dimension of the mean-pooled sentence embeddings. While cosine similarity is the dominant choice for embedding-based comparisons in NLP, we incorporate Pearson correlation as an experimental measure to assess the degree of linear alignment between the dimensions of the embeddings. This metric captures both directional alignment and consistency of magnitude changes across dimensions.

SemScore Instead of averaging token embeddings, this method uses the embedding of the special `[CLS]` token from the encoder output of a Transformer model as the sentence representation. The cosine similarity between the `[CLS]` embeddings of the student and teacher solutions is computed. This approach provides a focused and condensed semantic similarity assessment and is particularly effective when using encoder-based models such as BERT.

BERTScore This metric computes pairwise cosine similarity between all token embeddings of the student and teacher solutions and applies a token-matching algorithm to derive Precision, Recall, and F1 scores. We use the F1 score as the final semantic similarity measure, reflecting fine-grained alignment at the token level. This method leverages the bidirectional attention mechanism of encoder-based models to deliver high alignment accuracy.

Applicability of Metrics by Model Architecture We carefully matched each semantic similarity metric to the appropriate LLM architecture, taking into account both the intended design of the metrics and the structural characteristics of the models.

- Mean-pooled Embedding Cosine Similarity, Mean-pooled Embedding Pearson Correlation, and SemScore involve converting the full sentence into a fixed-length embedding vector and computing similarity between vectors. Decoder-based language models (e.g., GPT variants) operate in an auto-regressive fashion and can produce sentence-level embeddings by pooling over the final hidden states or using the last token representation. These metrics are thus compatible with decoder-based models for evaluating sentence-level semantic similarity.

- BERTScore, on the other hand, does not compress the sentence into a single embedding. Instead, it computes semantic alignment based on token-level similarity and optimal matching. This method is most effective with encoder-based models (e.g., BERT), which utilize bidirectional attention to model rich contextual dependencies across tokens. In contrast, auto-regressive decoding in GPT-like models limits such context integration, making them less suitable for BERTScore-based evaluation.

Accordingly, we selected and applied similarity metrics in alignment with the architectural characteristics of the LLMs to ensure that each metric could function as intended. The similarity computation procedures were consistently applied across all experiments, contributing to a broad and comprehensive exploration of semantic alignment methods.

E Preliminary Study Motivating Task Separation between GTA and SED

In this appendix, we presents preliminary experimental results for a simple integration of GTA outputs into SED. Our study is predicated on the hypothesis that GTA, which aims at alignment of reasoning flows, and SED, which targets precise detection of error occurrence points, are inherently distinct tasks in terms of educational objectives and evaluation criteria. Based on this hypothesis, the two tasks were designed and implemented independently. To assess the validity of this design choice, we constructed a setting in which GTA outputs were provided as additional inputs to SED (denoted as SED + GTA) and compared this configuration with the proposed SED standalone setting (SED, Ours).

Table 12 presents a comparative analysis of the SED standalone configuration and the GTA-conditioned configuration across English and Korean datasets. Overall, the inclusion of GTA outputs as additional inputs did not lead to consistent performance improvements over the SED standalone setting. In fact, performance degradation was observed across multiple models and language settings. Notably, even high-performing models such as GPT-4o, Claude-Sonnet-4, and Gemini-2.5-Flash exhibited lower error detection accuracy under the SED + GTA configuration. Some smaller models showed limited gains under specific language settings, but these effects were inconsistent across models and languages.

Model	SED (Ours)		SED + GTA	
	EN	KO	EN	KO
GPT-4o	94.4	95.8	46.2	61.8
Claude-Sonnet-4	87.8	90.2	25.0	32.4
Gemini-2.5-Flash	90.4	91.4	64.6	56.4
Llama-3.1-8B-Instruct	80.4	75.2	58.2	48.8
DeepSeek-R1-Distill-Llama-8B	55.4	57.8	28.6	34.2
DeepSeek-llama3.1-Blossom-8B	63.2	62.0	30.8	34.0
Qwen2.5-7B-Instruct	79.4	86.0	47.4	69.6
DeepSeek-R1-Distill-Qwen-7B	82.8	80.4	63.2	59.4
DeepSeek-R1-Distill-Qwen-7B-Multilingual	86.4	83.4	69.6	66.2
Mistral-7B-Instruct-v0.3	67.0	67.0	39.8	51.8
Phi-4-mini-instruct	76.6	78.6	43.6	53.2

Table 12: Preliminary results of integrating GTA outputs into SED. We compare the proposed SED module (left) with a variant that takes GTA outputs as input (right) on English and Korean datasets.

Further analysis indicates that while the reasoning-flow alignment information provided by GTA is useful for analyzing strategic similarities in student solutions, it does not provide the unambiguous error signals required by SED. In particular, GTA outputs may include placeholders (e.g., _) when steps in the student solution cannot be aligned with the reference solution. Such placeholders can introduce ambiguity that hinders SED from accurately identifying the initial error point. Even when subsequent steps partially align with the reference solution, GTA outputs may blur the boundaries of error occurrence or partially justify incorrect reasoning steps. This phenomenon can be interpreted as arising from the difference in task objectives and evaluation criteria between GTA, which emphasizes reasoning-flow alignment, and SED, which focuses on the precise detection of error points.

This trend persisted even in settings with relatively high-quality GTA outputs, suggesting that the observed performance degradation is not due to the intrinsic quality of GTA outputs but rather stems from a mismatch in the types of information and evaluation criteria required by the two tasks.

Model	**Baseline**		**Ours**	
	EN	**KO**	**EN**	**KO**
GPT-4o	75.2	72.6	**94.4**	**95.8**
Claude-sonnet-4	72.6	75.0	87.8	90.2
Gemini-2.5-Flash	**77.6**	**80.0**	90.4	91.4
Llama-3.1-8B-Instruct	42.2	**48.4**	80.4	75.2
DeepSeek-R1-Distill-Llama-8B	30.6	19.4	55.4	57.8
DeepSeek-llama3.1-Bllossom-8B	33.4	28.2	63.2	62.0
Qwen2.5-7B-Instruct	49.0	41.6	79.4	**86.0**
DeepSeek-R1-Distill-Qwen-7B	59.2	43.2	82.8	80.4
DeepSeek-R1-Distill-Qwen-7B-Multilingual	57.4	43.8	**86.4**	83.4
Mistral-7B-Instruct-v0.3	10.2	4.8	67.0	67.0
Phi-4-mini-instruct	**60.4**	43.8	76.6	78.6

Table 13: Accuracy comparison between Baseline and Ours for English (EN) and Korean (KO)

In other words, while GTA and SED inherently provide complementary information, a naive combination within the same input space can confound the decision criteria of each task.

Taken together, these preliminary results support the conclusion that designing and operating GTA and SED as independent tasks is more appropriate for preserving the educational objectives of each task and maintaining the reliability of error detection performance analyses. Future work should explore integration strategies that systematically leverage the complementary information of the two tasks without compromising task-specific evaluation criteria.

F Solution Error Detection without Answer Solutions

In this appendix, we presents additional experimental results aimed at evaluating the impact of providing reference (teacher) solutions on the performance of LLMs in error detection tasks.

One of the primary objectives of this study is to quantitatively assess whether LLMs can more accurately identify errors in student-generated solutions when accompanied by corresponding teacher solutions. The rationale for including teacher solutions in the proposed prompting strategy is to furnish the model with essential contextual grounding, thereby facilitating more precise localization and interpretation of student errors.

To empirically validate this hypothesis, we conducted an ablation study using the same test set but removed teacher solutions from the input. This configuration serves as a baseline to evaluate the model's ability to detect errors based solely on the student solution, without external reference.

As shown in Table 13, the inclusion of teacher solutions led to substantial improvements in error detection accuracy across most models. The performance gains were particularly pronounced for smaller-scale models (e.g., 7B, 8B), suggesting that teacher solutions offer valuable structural guidance to models with limited reasoning capacity.

Moreover, in domains such as mathematics—where complex, multi-step logical reasoning is essential—accurately identifying errors based solely on student output is often infeasible. When reference solutions are provided, LLMs can go beyond shallow textual similarity and instead evaluate logical consistency and correctness with greater precision. These quantitative results offer empirical support for the effectiveness of our proposed alignment-based error detection framework.

In this appendix, we presents additional experimental results aimed at evaluating the impact of providing answer solutions on the performance of LLMs in error detection tasks.

G Human Evaluator Guidelines for Solution Alignment

This appendix details the annotation guidelines and considerations that human raters followed when aligning student solutions with teacher solutions in the Bi-GSM8K dataset.

- **Ignore Arithmetic Mistakes:** Minor computational errors are disregarded. If the reasoning process or solution approach is logically aligned, the corresponding steps are matched.

- **Match Based on Logical Structure:** Steps are aligned based on logical structure, even if numerical values or surface expressions differ.

- **Marking Unmatched Steps:** When no appropriate matching step exists between the student and teacher solutions, annotators denote the step with an underscore (_) to indicate the absence of alignment. Examples are illustrated in Table 14 and Table 15.

- **One-to-One Matching Principle:** All alignments must follow a one-to-one correspondence between steps in the student and teacher solutions.

- **Handling Multi-Step Sentences:** When a single sentence in either solution contains multiple logical steps, annotators align it to the most salient or earliest relevant step based on semantic content.
  - For instance, if the teacher solution consists of two steps, "First, compute the total," followed by "Then, divide the result," but the student expresses both in a single sentence, such as "We can divide the total number," annotators determine which teacher step is more directly addressed and align accordingly.
  - Similarly, when a student step encapsulates multiple teacher steps, it is matched to the step that most semantically corresponds to its core meaning.

These guidelines are designed to support consistent and objective annotations while allowing annotators the flexibility to apply informed judgment, focusing on the logical and semantic alignment between steps.

Based on these alignment decisions, we assign a score of 1 to each aligned step and a score of 0 to unmatched or irrelevant steps, following the same evaluation scheme as the GTA module. The final similarity score is computed by summing all alignment scores and dividing by the total number of steps, yielding a normalized score between 0 and 1.

H Analysis of Human Evaluator Consistency

In this appendix, we quantitatively analyze the consistency of solution alignment results produced by five human evaluators involved in this study.

Each evaluator directly performed the alignment between student and teacher solutions by considering both the semantic similarity and logical flow at each step. Under subjective judgment, evaluators selected the most natural alignments, leaving unmatched steps blank when alignment was difficult.

The consistency of these multi-evaluator alignment results was measured using Fleiss' Kappa (McHugh, 2012), a statistical metric for assessing agreement among multiple raters on categorical data. Fleiss' Kappa is well-suited for our analysis as it objectively computes inter-rater agreement while correcting for chance agreement. The Kappa value ranges from –1 to 1, with values closer to 1 indicating higher agreement. The Fleiss' Kappa calculated in this study was 0.6149, demonstrating substantial agreement among the evaluators. Conventionally, Kappa values above 0.6 indicate reasonably strong consistency, supporting the reliability of the obtained evaluation results.

These findings provide objective evidence for the reliability and consistency of the similarity scores derived from human alignment and substantiate the validity of the quantitative evaluation method proposed in this work.

I Detailed Prompt Template for Solution Error Detection

In this appendix, we presents the detailed prompt template used in the SED module. As described in the main text, the SED module adopts a 3-shot prompting approach, as illustrated in Figure 4.

The three few-shot examples were carefully selected from the training data to evenly represent errors occurring at different reasoning stages (i.e., stages 1, 2, and 3). Each example consists of a problem statement and solution, step-by-step teacher solution, a student solution containing an error, and a query instructing the model to identify the first step at which the error occurs.

This prompt structure is designed to guide the LLM in comprehending the student's reasoning trajectory and solution logic in a stepwise manner, thereby enabling accurate identification of the initial point of error.

You are given a math problem, a correct answer solution, and a student solution. Your task is to compare the correct solution to the student solution and output the step number where the student's error begins. If the student's solution is completely correct, output 0.

Problem: Sohee feels bored with her current game and decides to play a new one. In the new game, 80% of the 100 hours of gameplay consists of repetitive and boring stages. However, through an expansion pack, she can add 30 hours of enjoyable stages. Including the expansion pack, how many hours of enjoyable stages can Sohee play?

Answer Solution:

```
{
    "step_1":"There are 100×0.8=«100×0.8=80»80 hours of boring stages in the game.",
    "step_2":"The enjoyable gameplay time is 100-80=«100-80=20»20 hours.",
    "step_3":"With the expansion pack, the enjoyable gameplay time increases to 20+30=«20+30=50»50 hours."
}
```

Student Solution:

```
{
    "step_1": "There are 100×0.8=«100×0.8=80»80 hours of boring stages in the game.",
    "step_2":"The enjoyable gameplay time is 100-80=«100-80=20»20 hours.",
    "step_3":"The expansion pack has 30×0.8=«30×0.8=24»24 hours of boring stages.",
    "step_4":"The enjoyable gameplay time in the expansion pack is 30-24=«30-24=6»6 hours.",
    "step_5":"The total enjoyable gameplay time is 50+6=«50+6=56»56 hours."
}
```

Q: Is the Student Solution incorrect? Write only the step number with the first error or 0 if no error is found.
A: 3

Problem: Seongjin is stranded on a deserted island. He needs salt to season fish. He collected 2 liters of seawater in an old bucket. If the water contains 20% salt, how many milliliters of salt will Seongjin get when all the water evaporates?

Answer Solution:

```
{
    "step_1":"First, find out how many liters of the seawater is salt: 2L * 20% = «2*0.2=0.4»0.4L",
    "step_2":"Then, multiply this amount by 1000 ml/L to find how much salt Seongjin gets: 0.4L * 1000ml/L = «0.4*1000=400»400ml"
}
```

Student Solution:

```
{
    "step_1":"220 = «220=40»40ml of salt is obtained."
}
```

Q: Is the Student Solution incorrect? Write only the step number with the first error or 0 if no error is found.
A: 1

Problem: A set of hairpins costs 3,000 won each, and a comb costs 1,000 won each. Jinri buys one set of hairpins and one comb. Sujeong buys three sets of hairpins and one comb. How much do the two girls spend in total?

Answer Solution:

```
{
    "step_1":"Jinri spends 3,000 + 1,000 = «3000+1000=4000»4,000 won.",
    "step_2":"The cost of three sets of hairpins is 3,000 x 3 = «3000*3=9000»9,000 won.",
    "step_3":"Sujeong spends 9,000 + 1,000 = «9000+1000=10000»10,000 won.",
    "step_4":"So the two girls spend 4,000 + 10,000 = «4000+10000=14000»14,000 won in total."
}
```

Student Solution:

```
{
    "step_1":"Jinri spends 3,000 + 1,000 = «3000+1000=4000»4,000 won.",
    "step_2":"The cost of two sets of hairpins is 3,000 x 2 = «3000*2=6000»6,000 won.",
    "step_3":"Sujeong spends 6,000 + 1,000 = «6000+1000=7000»7,000 won.",
    "step_4":"So the two girls spend 4,000 + 7,000 = «4000+7000=11000»11,000 won in total."
}
```

Q: Is the Student Solution incorrect? Write only the step number with the first error or 0 if no error is found.
A: 2

Problem: `{problem}`
Answer Solution: `{answer_solution}`
Student Solution: `{student_solution}`
Q: Is the Student Solution incorrect? Write only the step number with the first error or 0 if no error is found.
A:

Figure 4: Detailed Prompt Template for Solution Error Detection.

J Detailed Examples of Ground Truth Alignment

In this appendix, we present illustrative examples of GTA results for both English (Table 14) and Korean (Table 15). Each table includes the math problem ("Problem"), the step-by-step teacher solution ("Answer Solution"), and the student's step-by-step reasoning ("Student Solution"). The "Reference Alignment" shows gold-standard step-level alignments manually annotated by a human rater. Predicted alignments from the best-performing open-source model (BERTScore with `bert-large-uncased` and NW algorithm) and the top commercial LLM API (GPT-4o) are also provided, with misalignments highlighted in red.

Problem
Yeona has a 2L water bottle next to her desk. She takes a sip every 5 minutes, and each sip is 40ml. How many minutes does it take to finish one bottle of water?

Answer Solution
Step 1: First, find the total ml of the bottle: 2L * 1000ml/L = «2*1000=2000»2000ml Step 2: Then divide the total ml by the amount consumed per sip: 2000ml / 40ml = «2000/40=50»50 sips. Step 3: Then, multiply the number of sips by the time per sip to find the time it takes to drink the bottle: 50 sips * 5 minutes/sip = «50*5=250»250 minutes.

Student Solution
Step 1: Yeona's water bottle is 200ml. Step 2: Divide 200ml by the amount consumed per sip. 200ml / 40ml = «200/40=5»5 sips. Step 3: To find the time it takes to drink the bottle, multiply the number of sips by the time per sip: 5 sips * 5 minutes/sip = 25 minutes.

Reference Alignment	
Student Solution	**Answer Solution**
_	First, find the total ml of the bottle: 2L * 1000ml/L = «2*1000=2000»2000ml
Yeona's water bottle is 200ml.	_
Divide 200ml by the amount consumed per sip. 200ml / 40ml = «200/40=5»5 sips.	Then divide the total ml by the amount consumed per sip: 2000ml / 40ml = «2000/40=50»50 sips.
To find the time it takes to drink the bottle, multiply the number of sips by the time per sip: 5 sips * 5 minutes/sip = 25 minutes.	Then, multiply the number of sips by the time per sip to find the time it takes to drink the bottle: 50 sips * 5 minutes/sip = «50*5=250»250 minutes.

Predicted Alignment by BERTScore + bert-large-uncased + NW algorithm	
Student Solution	**Answer Solution**
Yeona's water bottle is 200ml.	First, find the total ml of the bottle: 2L * 1000ml/L = «2*1000=2000»2000ml
Divide 200ml by the amount consumed per sip. 200ml / 40ml = «200/40=5»5 sips.	Then divide the total ml by the amount consumed per sip: 2000ml / 40ml = «2000/40=50»50 sips.
To find the time it takes to drink the bottle, multiply the number of sips by the time per sip: 5 sips * 5 minutes/sip = 25 minutes.	Then, multiply the number of sips by the time per sip to find the time it takes to drink the bottle: 50 sips * 5 minutes/sip = «50*5=250»250 minutes.

Predicted Alignment by GPT-4o	
Student Solution	**Answer Solution**
_	First, find the total ml of the bottle: 2L * 1000ml/L = «2*1000=2000»2000ml
Yeona's water bottle is 200ml.	_
Divide 200ml by the amount consumed per sip. 200ml / 40ml = «200/40=5»5 sips.	Then divide the total ml by the amount consumed per sip: 2000ml / 40ml = «2000/40=50»50 sips.
To find the time it takes to drink the bottle, multiply the number of sips by the time per sip: 5 sips * 5 minutes/sip = 25 minutes.	Then, multiply the number of sips by the time per sip to find the time it takes to drink the bottle: 50 sips * 5 minutes/sip = «50*5=250»250 minutes.

Table 14: Example Results of GTA using GPT-4o and BERTScore with `bert-large-uncased` and NW Algorithm (English).

Problem
연아는 책상 옆에 2L짜리 물병을 두고 있습니다. 5분마다 한 모금씩 마시는데, 한 모금당 물의 양은 40ml입니다. 물 한 병을 다 마시는 데 몇 분이 걸리나요?

Answer Solution
Step 1: 먼저 병의 총 ml 수를 찾습니다: 2L * 1000ml/L = «2*1000=2000»2000ml Step 2: 그런 다음 총 ml 수를 한 모금당 마시는 양으로 나눕니다: 2000ml / 40ml = «2000/40=50»50 모금 Step 3: 그런 다음 모금 횟수에 모금당 시간을 곱하여 병을 마시는 데 걸리는 시간을 찾습니다: 50 모금 * 5분/모금 = «50*5=250»250분

Student Solution
Step 1: 연아의 물병은 200ml입니다. Step 2: 200ml를 한 모금당 마시는 양으로 나눕니다: 200ml / 40ml = «200/40=5»5 모금 Step 3: 모금 횟수에 모금당 시간을 곱하여 병을 마시는 데 걸리는 시간을 찾습니다: 5 모금 * 5분/모금 = «5*5=25»25분

Reference Alignment	
Student Solution	**Answer Solution**
_	먼저 병의 총 ml 수를 찾습니다: 2L * 1000ml/L = «2*1000=2000»2000ml
연아의 물병은 200ml입니다.	_
200ml를 한 모금당 마시는 양으로 나눕니다: 200ml / 40ml = «200/40=5»5 모금	그런 다음 총 ml 수를 한 모금당 마시는 양으로 나눕니다: 2000ml / 40ml = «2000/40=50»50 모금
모금 횟수에 모금당 시간을 곱하여 병을 마시는 데 걸리는 시간을 찾습니다: 5 모금 * 5분/모금 = «5*5=25»25분	그런 다음 모금 횟수에 모금당 시간을 곱하여 병을 마시는 데 걸리는 시간을 찾습니다: 50 모금 * 5분/모금 = «50*5=250»250분

Predicted Alignment by BERTScore + bert-large-uncased + NW algorithm	
Student Solution	**Answer Solution**
연아의 물병은 200ml입니다.	먼저 병의 총 ml 수를 찾습니다: 2L * 1000ml/L = «2*1000=2000»2000ml
200ml를 한 모금당 마시는 양으로 나눕니다: 200ml / 40ml = «200/40=5»5 모금	그런 다음 총 ml 수를 한 모금당 마시는 양으로 나눕니다: 2000ml / 40ml = «2000/40=50»50 모금
모금 횟수에 모금당 시간을 곱하여 병을 마시는 데 걸리는 시간을 찾습니다: 5 모금 * 5분/모금 = «5*5=25»25분	그런 다음 모금 횟수에 모금당 시간을 곱하여 병을 마시는 데 걸리는 시간을 찾습니다: 50 모금 * 5분/모금 = «50*5=250»250분

Predicted Alignment by GPT-4o	
Student Solution	**Answer Solution**
_	먼저 병의 총 ml 수를 찾습니다: 2L * 1000ml/L = «2*1000=2000»2000ml
연아의 물병은 200ml입니다.	_
200ml를 한 모금당 마시는 양으로 나눕니다: 200ml / 40ml = «200/40=5»5 모금	그런 다음 총 ml 수를 한 모금당 마시는 양으로 나눕니다: 2000ml / 40ml = «2000/40=50»50 모금
모금 횟수에 모금당 시간을 곱하여 병을 마시는 데 걸리는 시간을 찾습니다: 5 모금 * 5분/모금 = 25분	그런 다음 모금 횟수에 모금당 시간을 곱하여 병을 마시는 데 걸리는 시간을 찾습니다: 50 모금 * 5분/모금 = «50*5=250»250분

Table 15: Example Results of GTA using GPT-4o and BERTScore with bert-large-uncased and NW Algorithm (Korean)

K Detailed Examples of Solution Error Detection

This appendix provides illustrative examples of SED results on student solutions proposed in this paper. In Tables 16 and 17, "Problem" denotes the question prompt, "Answer Solution" refers to the teacher's solution, and "Student Answer" represents the student's solution. The "Answer" indicates the ground-truth location of the first error in the student's solution, while "Prediction" corresponds to the initial error point predicted by each model. Incorrect segments are highlighted in red, and correct segments are marked in blue.

Section	**Content**
Problem	Three cats were sitting on a fence meowing at the moon. The first cat meowed 3 times per minute. The second cat meowed twice as often as the first cat. The third cat meowed at 1/3 the frequency of the second cat. What is the total number of times the three cats meowed in 5 minutes?
Answer Solution	**Step 1:** The second cat meowed twice as often as the first cat, which meowed 3 times per minute, resulting in a total of $2 \times 3 = \boxed{6}$ meows per minute. **Step 2:** The third cat meowed at $\frac{1}{3}$ the frequency of the second cat, resulting in a total of $6 \div 3 = \boxed{2}$ meows per minute. **Step 3:** Therefore, the three cats meow $3 + 6 + 2 = \boxed{11}$ times per minute. **Step 4:** In 5 minutes, three cats meow $5 \times 11 = \boxed{55}$ times.
Student Solution	**Step 1:** The second cat meows $2 \times 3 = \boxed{6}$ times. **Step 2:** The third cat meows $3 \times \frac{1}{3} = \boxed{1}$ time. **Step 3:** The three cats meow $3 + 6 + 1 = \boxed{10}$ times per minute. **Step 4:** For 5 minutes, the three cats meow $10 \times 5 = \boxed{50}$ times.
Answer	Step 2
Prediction	**GPT-4o:** Step 2 **LLaMA-3.1-8B-Instruct:** Step 2 **DeepSeek-R1-Distill-LLaMA-8B:** Step 3 **DeepSeek-LLaMA3.1-Blossom-8B:** Step 3 **Qwen2.5-7B-Instruct:** Step 3 **DeepSeek-R1-Distill-Qwen-7B:** Step 1 **DeepSeek-R1-Distill-Qwen-7B-Multilingual:** Step 1 **Mistral-7B-Instruct-v0.3:** Step 3 **Phi-4-mini-instruct:** Step 2

Table 16: Example of SED Using GPT-4o and Open LLMs.(English) Blue indicates correct detection; red indicates incorrect detection.

Section	**Content**
Problem	고양이 세 마리가 울타리에 앉아 달을 향해 야옹거리고 있었습니다. 첫 번째 고양이는 분당 3번 야옹거렸습니다. 두 번째 고양이는 첫 번째 고양이보다 두 배 더 자주 야옹거렸습니다. 세 번째 고양이는 두 번째 고양이의 1/3 빈도로 야옹거렸습니다. 고양이 세 마리가 5분 동안 야옹거리는 총 횟수는 얼마입니까?
Answer Solution	**Step 1:** 두 번째 고양이는 첫 번째 고양이가 분당 3번 야옹하는 것보다 두 배 더 자주 야옹하여 분당 총 $2 \times 3 = \boxed{6}$번 야옹했습니다. **Step 2:** 세 번째 고양이는 두 번째 고양이의 1/3 빈도로 야옹하여 분당 총 $6 \div 3 = \boxed{2}$번 야옹했습니다. **Step 3:** 따라서 세 마리 고양이는 분당 $3 + 6 + 2 = \boxed{11}$번 야옹합니다. **Step 4:** 5분 동안 세 고양이는 $5 \times 11 = \boxed{55}$번 야옹합니다.
Student Solution	**Step 1:** 두 번째 고양이는 $2 \times 3 = \boxed{6}$번 야옹합니다. **Step 2:** 세 번째 고양이는 $3 \times \frac{1}{3} = \boxed{1}$번 야옹합니다. **Step 3:** 세 고양이는 분당 $3 + 6 + 1 = \boxed{10}$번 야옹합니다. **Step 4:** 5분 동안 세 고양이는 $10 \times 5 = \boxed{50}$번 야옹합니다.
Answer	Step 2
Prediction	**GPT-4o:** Step 2 **LLaMA-3.1-8B-Instruct:** Step 2 **DeepSeek-R1-Distill-LLaMA-8B:** Step 3 **DeepSeek-llama3.1-Blossom-8B:** Step 3 **Qwen2.5-7B-Instruct:** Step 2 **DeepSeek-R1-Distill-Qwen-7B:** Step 2 **DeepSeek-R1-Distill-Qwen-7B-Multilingual:** Step 2 **Mistral-7B-Instruct-v0.3:** No errors **Phi-4-mini-instruct:** Step 2

Table 17: Example of SED Using GPT-4o and Open LLMs.(Korean) Blue indicates correct detection; red indicates incorrect detection.

L Visualization of GTA and SED Experimental Results

This appendix presents visualizations of model performance and inference latency for the two core tasks discussed in the main text: **GTA** and **SED**.

L.1 Visualization of Model Performance

For the GTA task, model performance is depicted using radar charts illustrating Pearson correlation in Figures 5 and Spearman correlation in Figures 6. For the SED task, model performance is shown through a radar chart of accuracy in Figure 7.

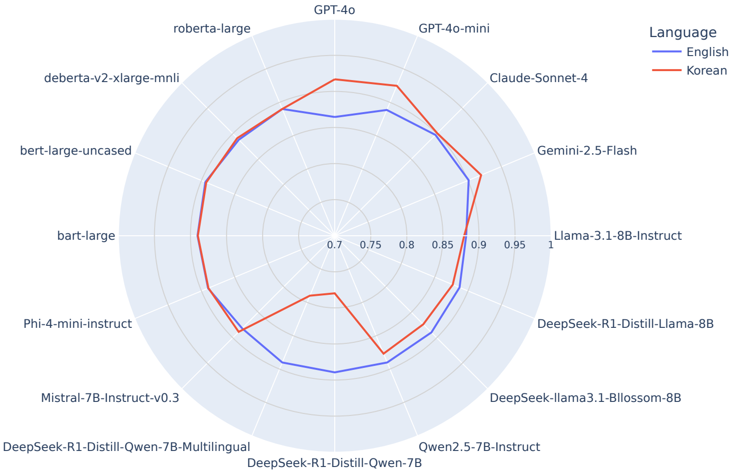

Figure 5: **Ground Truth Alignment Performance (Pearson Correlation)**. Radar plot illustrating model-wise Pearson correlation scores. Blue lines represent performance on English tasks, while orange lines indicate performance on Korean tasks. This visualization reveals language-specific variation in alignment capabilities across models.

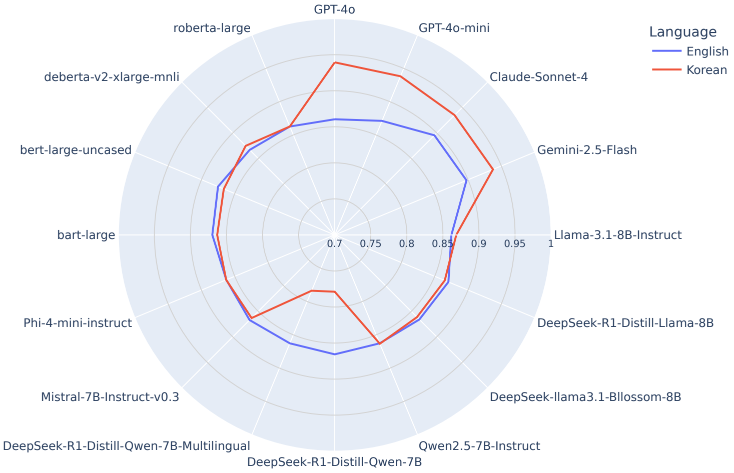

Figure 6: **Ground Truth Alignment Performance (Spearman Correlation)**. Radar plot illustrating model-wise Spearman correlation scores. Blue lines represent performance on English tasks, while orange lines indicate performance on Korean tasks. This visualization reveals language-specific variation in alignment capabilities across models.

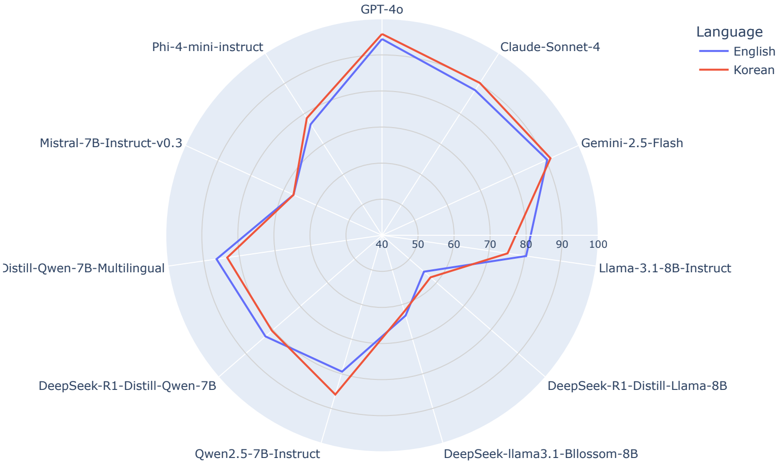

Figure 7: **Solution Error Detection Performance (Accuracy)**. Radar plot showing accuracy scores for each model on the SED task. Blue lines represent English task performance, and orange lines represent Korean task performance. The figure highlights differences in multilingual generalization and model robustness in error identification.

L.2 Visualization of Model Latency

To provide a more detailed understanding of model efficiency, Figures 8 and 9 present the inference latency (in seconds) of each model for the respective tasks. All latency values represent averages across English and Korean experiments per model and were measured under consistent runtime environments.

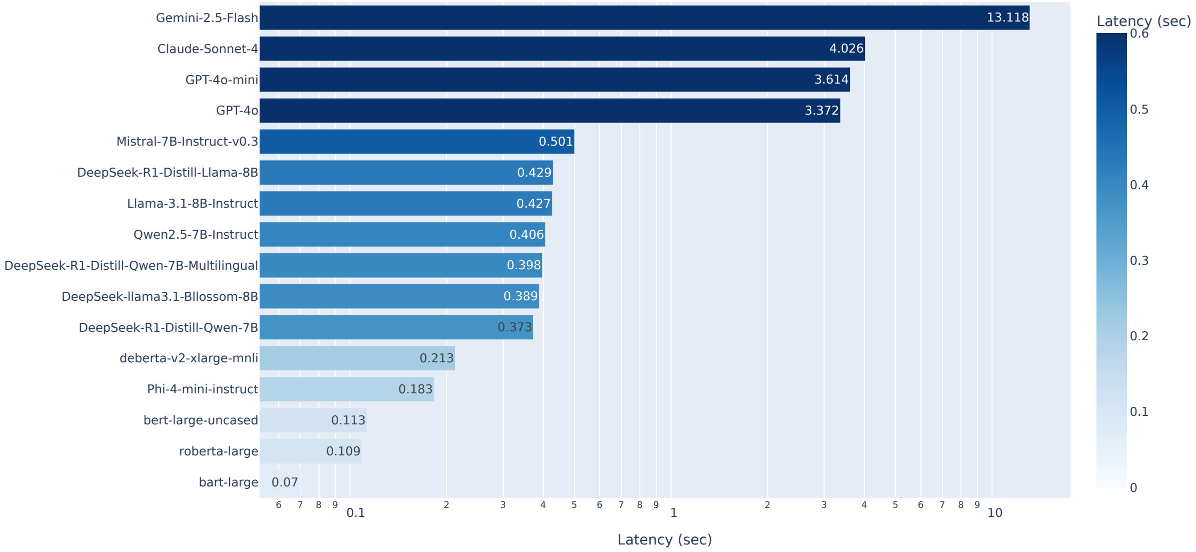

Figure 8: **Latency on Ground Truth Alignment Task**. Latency (in seconds) per model, averaged over both English and Korean examples. Lower latency reflects faster inference. This comparison enables assessment of the trade-off between alignment quality and computational cost.

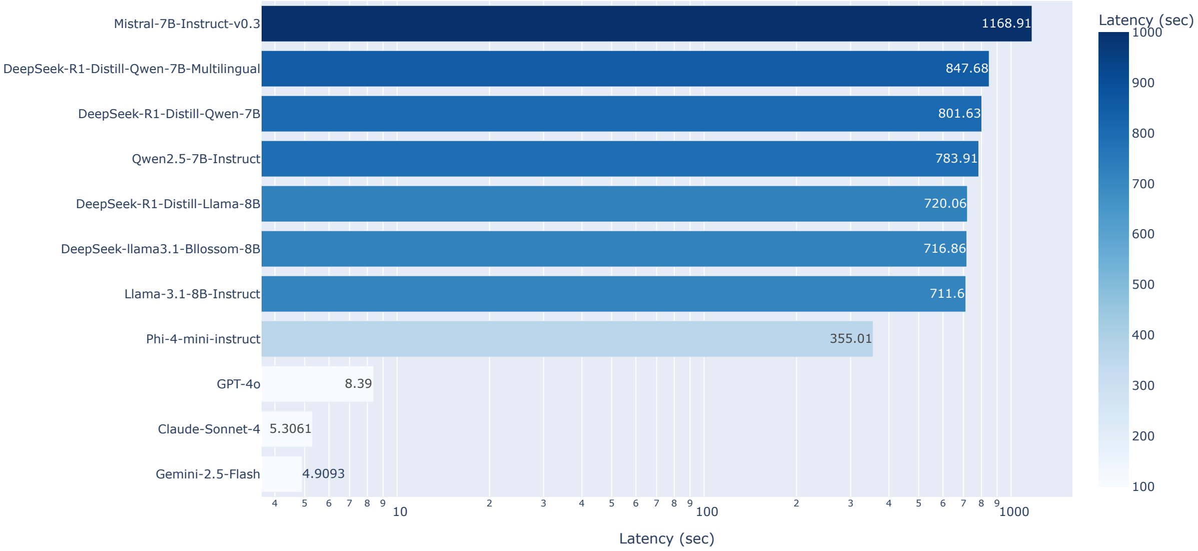

Figure 9: **Latency on Solution Error Detection Task**. Latency (in seconds) per model, averaged over English and Korean data. Lower latency implies more efficient error detection. The figure provides insight into the computational scalability of each model for educational feedback applications.