VEHME: A Vision-Language Model For Evaluating Handwritten Mathematics Expressions

Thu Phuong Nguyen[♠*] **Duc M. Nguyen**[◇*]
Hyotaek Jeon[◇] **Hyunwook Lee**[◇] **Hyunmin Song**[♠]
Sungahn Ko[◇†] **Taehwan Kim**[♠†]
[♠]Ulsan National Institute of Science and Technology
[◇]Pohang University of Science and Technology
{phuongnt, hyunminsong, taehwankim}@unist.ac.kr
{ducnm, taek98, hwlee0916, sungahn}@postech.ac.kr

Abstract

Automatically assessing handwritten mathematical solutions is an important problem in educational technology with practical applications, but it remains a significant challenge due to the diverse formats, unstructured layouts, and symbolic complexity of student work. To address this challenge, we introduce VEHME-a **V**ision-Language Model for **E**valuating **H**andwritten **M**athematics **E**xpressions—designed to assess open-form handwritten math responses with high accuracy and interpretable reasoning traces. VEHME integrates a two-phase training pipeline: (i) supervised fine-tuning using structured reasoning data, and (ii) reinforcement learning that aligns model outputs with multi-dimensional grading objectives, including correctness, reasoning depth, and error localization. To enhance spatial understanding, we propose an Expression-Aware Visual Prompting Module, trained on our synthesized multi-line math expressions dataset to robustly guide attention in visually heterogeneous inputs. Evaluated on AIHub and FERMAT datasets, VEHME achieves state-of-the-art performance among open-source models and approaches the accuracy of proprietary systems, demonstrating its potential as a scalable and accessible tool for automated math assessment. Our training and experiment code is publicly available at our GitHub repository.

1 Introduction

The assessment of handwritten mathematical solutions is a fundamental yet labor-intensive component of mathematics education, serving as both a diagnostic tool for educators and a critical feedback mechanism for learners (Scarlatos et al., 2025; Lin, 2024; Zhang et al., 2021; Nakamoto et al., 2023). In traditional classroom settings, teachers manually evaluate student work to identify conceptual misunderstandings, procedural errors, and gaps in logical reasoning. But, the scalability of this process is severely limited by time constraints, class sizes, and the cognitive load of interpreting diverse solution strategies (Gowda and Suma, 2017; Laws et al., 2003; Callahan et al., 2021). With the rise of digital learning platforms, there is an urgent need for automated systems that can replicate the nuanced judgment of human graders while accommodating the unstructured, creative, and often messy nature of student-written content (Baral et al., 2025; Scarlatos et al., 2025).

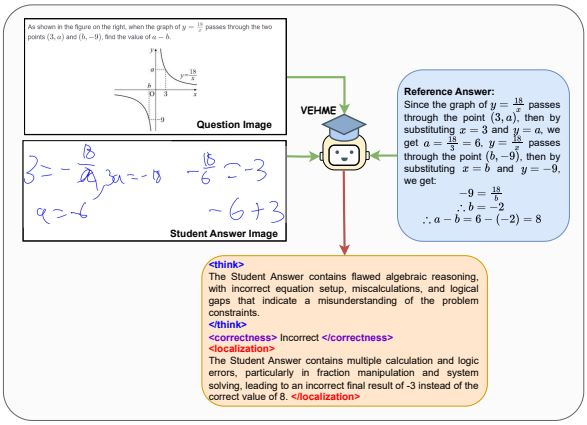

Figure 1: Our model takes a question, reference answer, and student answer image as input to predict the correctness of the student's solution and identify any error locations, if the solution is incorrect.

To address scalability challenges in grading, prior work has explored automated systems built around structured templates and Optical Character Recognition (OCR) pipelines (Li et al., 2019; Zhang et al., 2020, 2025; Wagstaff et al., 2019), as well as typed-text inputs processed by language models (Scarlatos et al., 2025). However, these approaches rely on rigid assumptions: that student responses are syntactically clean, spatially struc-

[*]Both authors contributed equally to this work.
[†]Co-corresponding authors.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 31793–31813
November 4-9, 2025

tured, or written within constrained templates. Furthermore, there is a notable lack of high-quality datasets that include complex, multi-expression handwritten math expressions (Yang et al., 2023), limiting the capacity to model open-form student solutions. Consequently, these systems often break down when applied to real-world data that is handwritten, unstructured, and heterogeneous, featuring symbolic math, diagrams, and natural language, all rendered in varied handwriting styles (Figure 1).

Previous approaches that leverage Large Language Models (LLMs) for grading often depend solely on the model's end-to-end reasoning capabilities without explicitly supervising or constraining the underlying reasoning process (Zhang et al., 2025; Mok et al., 2024; Kortemeyer et al., 2024). In the text-only domain, Process Reward Models (PRMs) have been introduced to address this issue by enabling step-by-step evaluation of mathematical sequential reasoning (He et al., 2024; Wang et al., 2024; Lambert et al., 2024). More recently, PRMs have been extended into the multimodal domain (Khalifa et al., 2025; Wang et al., 2025), allowing reasoning-aware supervision in visual tasks. However, to the best of our knowledge, there is still no effective approach tailored to handwritten mathematical expressions, a domain that combines the complexities of symbolic reasoning with the challenges of visual noise and layout diversity.

While recent advances in Vision-Language Models (VLMs) have shown promise in multimodal understanding, their application to handwritten math assessment has been limited by several key challenges: (C1) the scarcity of high-quality training data that pairs handwritten expressions with detailed error annotations (Jin et al., 2025), (C2) the difficulty in robustly processing the spatial layout and visual noise characteristic of student work (Guo et al., 2025b).

In this work, we take a step toward overcoming these challenges by tackling the underexplored and difficult task of grading template-free, handwritten open-response solutions in the broader domain of K–12 mathematics. This setting requires models capable of handling significant visual heterogeneity, unstructured layouts, and sequential reasoning, where early mistakes may propagate across later steps. To this end, we present **VEHME** (a **V**ision-Language Model for **E**valuating **H**andwritten **M**athematics **E**xpressions), a novel end-to-end framework for automatically grading handwritten math solutions. Our key contributions are summarized as follows:

- To address the challenge of assessing complex open-form mathematical expressions, we propose a novel framework named **VEHME**. To the best of our knowledge, we are the first to propose an end-to-end training pipeline for grading handwritten mathematical expressions.
- We adopt a dual-phase training regime: supervised fine-tuning on our synthesized datasets, followed by reinforcement learning with a composite reward to optimize for correctness, reasoning quality, and error localization.
- To address C1, we also propose a data synthesis pipeline to construct a high-quality reasoning-augmented dataset that pairs handwritten expressions with error detection and localization reasoning traces.
- To address C2, we propose the Expression-aware Visual Prompting Module (EVPM), equipping the VLM with precise spatial localization of handwritten mathematical expressions. To train this module, we construct a complex, heterogeneous, multi-expression handwritten math expressions dataset with its corresponding spatial information.
- Our training pipeline enables small, open-source Vision-Language Models (VLMs) to surpass the performance of larger open-source counterparts and achieve results on par with state-of-the-art proprietary models.

2 Related Work

2.1 Handwritten Mathematical Expression Recognition

Handwritten Mathematical Expression Recognition (HMER) has been a long-standing and evolving research area. The inherent ambiguity and complex structure in handwritten symbols present significant challenges. Traditional methods typically employed a two-stage process: mathematical symbol recognition followed by structure analysis (Chan and Yeung, 1998, 2000).

The advent of deep learning spurred a shift towards end-to-end OCR-based models. These approaches aimed to directly predict the symbolic

representation (e.g., LaTeX) from an input handwritten mathematical expression image. Variations included models that directly output the sequence (Deng et al., 2017; Li et al., 2022; Wu et al., 2020; Zhao and Gao, 2022) and others that incorporated grammatical rules of mathematical expressions to guide and constrain the prediction process (Li et al., 2024b; Wu et al., 2022, 2021).

More recently, Vision Language Models (VLMs), which combine vision encoders with Large Language Models (LLMs), have shown strong multimodal capabilities (Liu et al., 2023; Bai et al., 2023; Chen et al., 2024c). There is emerging interest in applying VLMs to HMER, leveraging their potential for understanding complex visual and textual relationships (Guo et al., 2025c).

2.2 Automatic Grading of Handwritten Math

The labor-intensive nature of manually grading handwritten mathematical solutions has driven the demand for automated systems. Early efforts in automated grading were often limited to structured or short-answer questions due to technological constraints. A prevalent traditional method involved converting handwritten expressions to a format like LaTeX using the HMER model, followed by evaluation with rule-based system or heuristics (Li et al., 2019; Zhang et al., 2020; Chaowicharat and Dejdumrong, 2023).

With the development of LLMs, new approaches have emerged where LLMs compare a student's formatted submission (derived via HMER) against a correct solution (Zhang et al., 2025). For finer evaluation (e.g., assessing complex solutions or providing step-level feedback), reinforcement learning (RL) has been explored to facilitate error localization and step-by-step correction (Li et al., 2025; Zheng et al., 2024). VLMs have also been investigated for direct automatic grading from handwritten input, aiming to bypass the explicit HMER formatting stage (Nath et al., 2025; Baral et al., 2025; Jin et al., 2025; Mok et al., 2024). However, the VLM-based systems show considerable sensitivity to handwriting quality, which currently restricts their robustness in practical, real-world scenarios.

2.3 Mathematical Reasoning and LLM as a Judge

The recent success of reasoning-focused models across domains such as mathematics and programming (OpenAI, 2024) draws significant attention to enhancing the reasoning capabilities of LLMs through reinforcement learning (RL) (Zhang et al., 2024b,a; Trung et al., 2024; Chen et al., 2024b). In this context, Guo et al. (2025a) introduced a novel RL-based approach that removes the need for supervised fine-tuning while achieving strong and reliable reasoning performance. Building on this progress, recent efforts have begun exploring similar techniques in multi-modal settings, attracting increasing research interest (Liu et al., 2025; Deng et al., 2025; Zhou et al., 2025).

In tasks requiring open-ended responses, such as error localization in educational contexts or code reviews, human evaluation is often necessary to validate correctness. However, this dependence on manual judgment limits scalability. Consequently, there is growing interest in using LLMs themselves as automated judges to evaluate the quality and accuracy of responses (Zheng et al., 2023; Chen et al., 2024a; Gu et al., 2024). Recent work has started to integrate LLM-based evaluators directly into RL training pipelines for such open-ended tasks.

3 Methodology

Our approach to Handwritten Mathematics Grading integrates two key components: (1) supervised fine-tuning using distilled data from the `QwQ-32B` model to instill foundational reasoning capabilities, and (2) reinforcement learning via Group Relative Policy Optimization (Shao et al., 2024) to refine the model's ability to generate accurate and explainable grading outputs. As shown in the Figure 2, this dual-phase training strategy aims to enhance both the correctness assessment and the interpretability of the model's evaluations.

3.1 Problem Formulation

We formulate the *Handwritten Mathematics Grading* problem as a vision-language modeling task, where the goal is to assess the correctness of a student's handwritten mathematical answer based on a given question and a reference solution.

Let each instance be represented by the tuple $x = (q, r, s)$, where: (i) $q \in \mathbb{P}^*$ is the **question**, represented as a sequence of pixels (e.g., an image of a printed or handwritten mathematical question), (ii) $r \in \mathbb{V}^*$ is the **reference answer**, expressed as a sequence of structured mathematical tokens from a vocabulary $\mathbb{V}$, and (iii) $s \in \mathbb{P}^*$ is the **student answer**, also represented as a sequence of pixels (i.e., a handwritten response). Here, $\mathbb{P}$ denotes the space of pixel values (e.g., grayscale or RGB), and

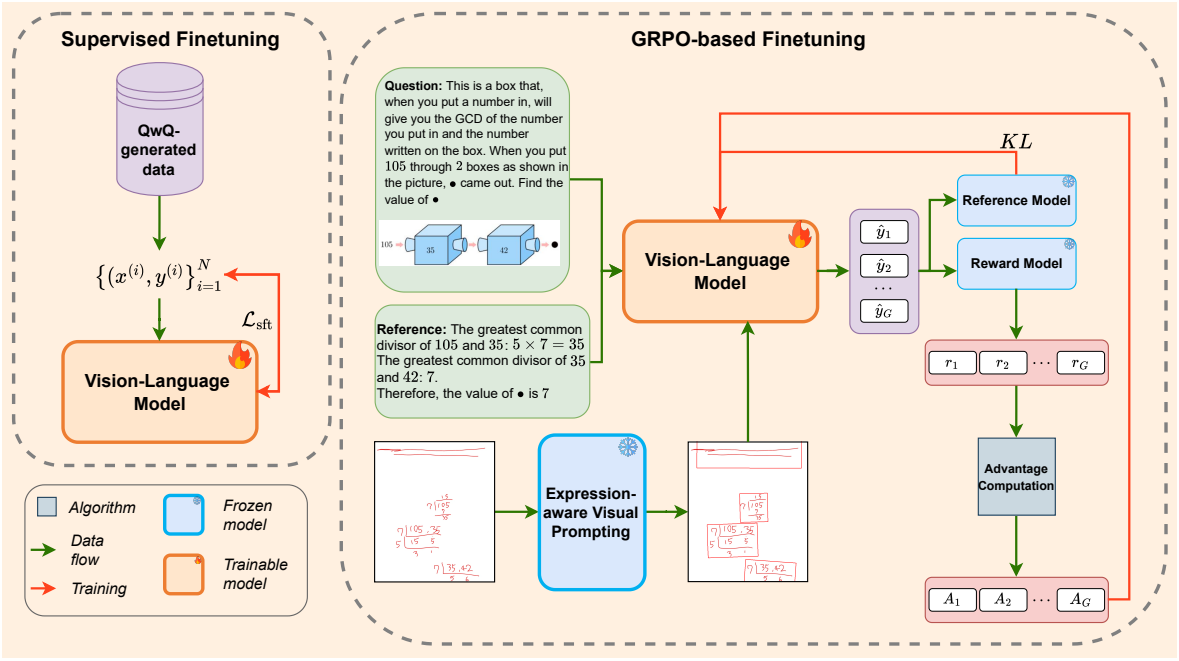

Figure 2: VEHME overview. Initially, a VLM is fine-tuned to generate outputs in the desired format using synthesized data from Sec 3.2.1. A subsequent Preference Optimization step trains the model using GRPO (Shao et al., 2024) method. This optimization is guided by rewards described in Sec 3.2.2 and takes the given problem, reference answer, and an expression-aware visual-prompted image from the student's answer as input.

$\mathbb{V}$ denotes the vocabulary of natural language and symbolic mathematical expressions (e.g., LaTeX or semantic tokens). Each instance is associated with a grading label $y = (c, l)$, where: (i) $c \in \{0, 1\}$ indicates the **correctness** of the student answer, and (ii) $l \in \mathbb{V}^*$ denotes the **location or description of the error**, provided as a (possibly empty) subsequence of tokens aligned to the student answer.

Let $\mathbb{X} = \mathbb{P}^* \times \mathbb{V}^* \times \mathbb{P}^*$ and $\mathbb{Y} = \{0, 1\} \times \mathbb{V}^*$ denote the input and output spaces, respectively. The goal is to learn a function $f : \mathbb{X} \to \mathbb{Y}$, that maps each input tuple $x = (q, r, s)$ to a grading label $y = (c, l)$, capturing both correctness and, if applicable, the symbolic location of the error in the student's response. To approximate f, we train a vision-language model π_θ parameterized by θ, which defines an autoregressive conditional distribution $\pi_\theta(y_t \mid x, y_{<t})$ over tokens in the output space. The function f is then approximated by $f(x) \approx \arg\max_{y \in \mathbb{Y}^*} \prod_{t=1}^{T} \pi_\theta(y_t \mid x, y_{<t})$, where T is the length of the output sequence y, and $y_{<t}$ denotes the previously generated tokens.

3.2 Main Techniques

3.2.1 Data Synthesis and Supervised Fine-tuning

Data synthesis from QwQ-32B Zheng et al. (2024) demonstrates that `QwQ-32B` (QwenTeam, 2025), a large language model built upon Qwen2.5-32B (Yang et al., 2024) and optimized specifically for mathematical reasoning, achieves state-of-the-art performance among open-source LLMs on step-level mathematical evaluation tasks, even surpassing fine-tuned process reward models (PRMs). This strong reasoning capability, despite `QwQ-32B`'s relatively modest size compared to models like GPT-4 (Achiam et al., 2023) or DeepSeek-R1 (Guo et al., 2025a), makes it a compelling teacher model for mathematical evaluation.

To capitalize on this, we perform supervised fine-tuning using data distilled from `QwQ-32B` as shown in the Figure 3. Given a tuple $x = (q, r, s)$ consisting of a question, a reference answer, and a student's handwritten response, we prompt `QwQ-32B` to assess the correctness of the answer and, if incorrect, to identify or describe the error. The model is instructed to generate responses in a structured format composed of three fields: a detailed thinking process, a correctness label, and an error localization. We set a maximum token budget of M for efficient training, as longer models' responses do not necessarily improve the model's performance (Chen et al., 2025b; Yeo et al., 2025; Deng et al., 2025; Chen et al., 2025a). In cases

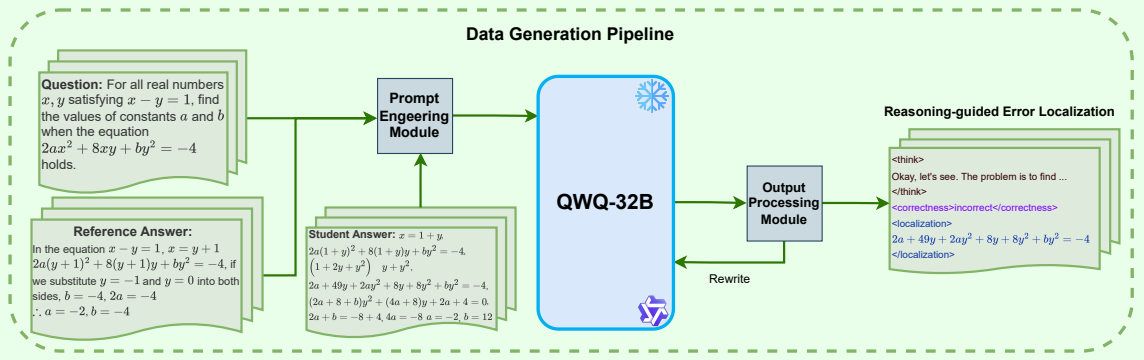

Figure 3: Process for synthesizing SFT data. `QwQ-32B` creates structured feedback from inputs *(q, r, s)* under a token budget *M*. Truncated outputs are repaired using a output processing module (Appendix A.2) with grammar-constrained decoding (Appendix A.2) to ensure valid SFT data.

where the model is truncated due to exceeding the token budget, we apply a structured post-processing routine that repairs truncated outputs and ensures conformance to the pre-defined format. Specifically, we truncate the response after the last complete thought from the model, append a closing `"</think>"` tag, and resume generation starting from `"<correctness>"` using grammar-constrained decoding (Geng et al., 2023). This allows the remainder of the response to be completed in a syntactically valid manner. The full post-processing procedure, the grammar, and more samples of the synthesized dataset can be found in Appendix A.2.

Supervised Fine-Tuning The distilled dataset from `QwQ-32B` provides high-quality demonstrations of expert-level reasoning and error localization. We use this data to initialize our model via supervised fine-tuning (SFT) (Figure 2), training it to mimic the teacher model's structured responses through next-token prediction.

3.2.2 Reinforcement Learning with Group Relative Preference Optimization

To improve the model's ability to produce accurate and explainable grading outputs, we adopt Group Relative Policy Optimization (GRPO) (Shao et al., 2024), a reinforcement learning technique that leverages intra-group comparisons among multiple completions generated per prompt, effectively optimizing the relative ranking of responses without relying on value function estimation. This approach has demonstrated significant improvements in mathematical reasoning tasks.

Recent studies have shown that GRPO facilitates stable training and emergent reasoning behavior, often surpassing supervised fine-tuning (Chen et al., 2025a; Deng et al., 2025; Zhou et al., 2025; Shen et al., 2025). Liu et al. (2025) further emphasizes GRPO's robustness, while Chen et al. (2025b) shows that judgment tasks, closely related to our grading tasks, are reasoning-intensive and benefit significantly from GRPO-based optimization.

GRPO formulation Formally, for each input x, we sample a group $G = \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_{|G|}\}$ of completions from the current policy π_θ. Each response $\hat{y}_i = (\hat{c}_i, \hat{l}_i)$ is evaluated using a composite function $r(\hat{y}_i)$. The advantage is efficiently estimated using the standardization of the rewards within the group: $\hat{A}_i = \frac{r(\hat{y}_i) - \mu_G}{\sigma_G}$, where μ_G and σ_G are the mean and standard deviation of the rewards in group G, respectively. The GRPO objective is then formulated similarly to Shao et al. (2024) as:

$$\mathcal{J}_{\text{GRPO}} = \mathbb{E}\big[x \sim P(X), \{y_i\}_{i=1}^{|G|} \sim \pi_{\theta_{\text{old}}}(y_i|x)\big]$$
$$\frac{1}{|G|}\sum_{i=1}^{|G|}\frac{1}{|\hat{y}_i|}\sum_{t=1}^{|\hat{y}_i|}\Big[\min\Big(\hat{A}_i\rho_{i,t}, \hat{A}_i\varsigma_{i,t}\Big) - \beta\mathcal{D}_\theta\Big],$$
$$\varsigma_{i,t} = \text{clip}\big(\rho_{i,t}, 1-\epsilon, 1+\epsilon\big),$$
$$\mathcal{D}_\theta = \mathbb{D}_{KL}\big[\pi_\theta \,||\, \pi_{\text{ref}}\big],$$

where $\rho_{i,t} = \frac{\pi_\theta(\hat{y}_{i,t}|x, \hat{y}_{i,<t})}{\pi_{\theta_{\text{old}}}(\hat{y}_{i,t}|x, \hat{y}_{i,<t})}$ is the probability ratio between the current and previous policies, $\mathbb{D}_{KL}\big[\pi_\theta||\pi_{\text{ref}}\big]$ is the unbiased estimator of the KL divergence between the current and the reference policies (Schulman et al., 2017), and ϵ, β are hyperparameters controlling the clipping range and the KL penalty (Shao et al., 2024).

Reward Modeling Each response is scored using a composite reward function:

$$r(\hat{y}_i) = r_{\text{match}}(\hat{y}_i) + r_{\text{loc}}(\hat{y}_i) + r_{\text{len}}(\hat{y}_i) + r_{\text{cos}}(\hat{y}_i) + r_{\text{rep}}(\hat{y}_i),$$

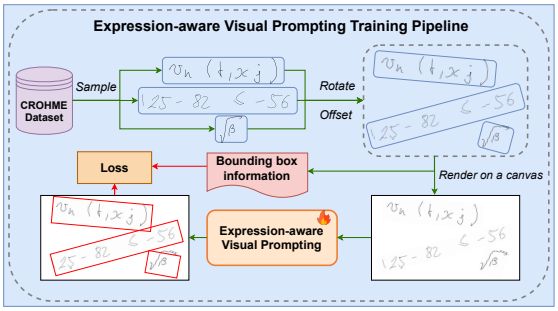

Figure 4: EVPM training pipeline. The model learns to predict ground truth bounding boxes for augmented mathematical expressions which are drawn onto a white backgrounds after augmentation.

where

$$r_{\text{match}}(\hat{y}_i) = \begin{cases} 1 & \text{if } \hat{c}_i = c_i \\ 0 & \text{otherwise} \end{cases},$$

$$r_{\text{loc}}(\hat{y}_i) = \begin{cases} 1 & \text{if } \hat{l}_i \text{ correctly describe the error} \\ 0 & \text{otherwise} \end{cases},$$

$$r_{\text{len}}(\hat{y}_i) = \begin{cases} 0.25 & \text{if } |\hat{y}_i| \geq 150 \\ 0 & \text{otherwise} \end{cases},$$

and $r_{\cos}, r_{\text{rep}}$ are the Cosine Reward and the repetition penalty (Yeo et al., 2025). For r_{loc}, traditional rule-based evaluation is inadequate due to the open-ended nature of the task and the variability in valid explanations. Prior work has demonstrated that QWQ-32B achieves great results in text-based step-level error localization (Zheng et al., 2024). Thus, we opt to use QWQ-32B as an automated judge to evaluate the correctness and quality of the model's error localization outputs for this component of the reward function.

3.2.3 Expression-aware Visual Prompting

Previous studies have shown that Vision-Language Models (VLMs) struggle to interpret multi-line handwritten mathematical expressions, which are common in students' solutions (Guo et al., 2025c). Recognizing these complex layouts is essential for downstream tasks such as automated grading. To address this, we propose an *Expression-aware Visual Prompting Module* (EVPM), which provides the model with spatial cues by generating oriented bounding boxes around individual mathematical expressions (Li et al., 2024a).

The EVPM comprises an oriented bounding box predictor trained on a synthetic dataset of multi-line mathematical expressions (Figure 4). To train the EVPM component, we utilize Yolov11 (Jocher et al., 2023) as the backbone of the bounding box detector. We construct the synthetic dataset by first sampling expressions from the grammar-based generator proposed by Truong et al. (2022), then rendering each expression as a handwritten image using symbol-level traces. Recognizing that students often write in non-linear or skewed patterns, particularly when unbounded by ruled lines, we simulate these realistic distortions through a stochastic layout pipeline. Each expression is randomly rotated and placed on a blank canvas with vertical spacing and padding variation, forming a more naturalistic setting for bounding box supervision. To create such expression-rich canvases, we formalize the image construction process in Appendix A.3. Each mathematical expression is first rendered into an image, then randomly rotated and positioned vertically on the canvas. The corresponding oriented bounding boxes are computed by rotating each image's axis-aligned box by the same angle. This process supports diverse and realistic handwriting layouts during training.

By training the EVPM on these synthetic examples, the model learns to identify individual expressions even under irregular layouts. This bounding box information is provided to the downstream VLM as visual prompts, enabling more accurate understanding of complex student solutions.

4 Experiments

4.1 Evaluation Datasets and Metrics

We use two datasets for handwritten mathematical assessment: the AIHub dataset[1] and the FERMAT dataset (Nath et al., 2025). More information about datasets' statistics and samples can be found in Appendix A.1.

AIHub dataset is derived from a large-scale educational repository encompassing K–12 mathematics problems. It comprises 183,085 handwritten student responses corresponding to 30,050 unique problem questions, originally authored in Korean. To mitigate potential performance degradation due to cross-lingual inputs, we filtered out student answers containing non-English characters. The remaining questions and reference answers were translated into English using the Google Translate API[2]. In the end, we obtained 81,394 training samples and 9,062 test samples for the AIHub dataset.

[1] https://aihub.or.kr

[2] https://cloud.google.com/translate/docs/reference/rest

FERMAT dataset provides a complementary evaluation focused on error analysis, so incorrect solutions significantly outnumber correct ones at an approximate ratio of 85:15, which is different from the balanced distribution in the filtered AIHub dataset. It contains 2,244 manually curated solutions across eight mathematical domains from arithmetic to calculus. Each entry contains: the original question, a handwritten solution image with intentional errors, the gold-standard correct answer, and error detection labels. Due to its small size, we opt not to perform any additional filtering, as doing so could lead to an insufficient amount of data for fine-tuning. We split the dataset into 70% training and 30% test sets.

Metrics For error detection performance evaluation, we employ two key metrics– **Accuracy**: measures overall prediction correctness across both error-present and error-free samples, and **F1-Score**: balances precision and recall to handle class imbalance, particularly crucial for FERMAT's skewed error distribution.

4.2 Implementation Details

We conduct our experiments using Qwen2.5-VL-7B-Instruct (Bai et al., 2025), a pre-trained large multimodal model. As an initial step, we perform supervised fine-tuning (SFT) using text-only data distilled from `QwQ-32B` (QwenTeam, 2025) to align the model with the desired output format. During reinforcement learning with GRPO, input images are resized to limit visual token overhead, capped at 50,176 pixels (224×224 image resolution) for the AIHub dataset and 501,760 pixels for FERMAT, preserving essential content while improving training efficiency. For more details, please refer to Appendix A.4.

4.3 Comparison with the State-of-the-Art

Experimental setup We conduct experiments on two subtasks of the mathematical grading task: Error Detection (ED) and Error Localization (EL), evaluating a cascaded setup where ED and EL are executed sequentially (Figure 5). The input for VLM processing consists of a problem question, a reference answer, and a student's solution. For the ED task, the model outputs a binary classification indicating whether the student's solution contains an error (Incorrect solution) or is error-free. Performance is evaluated based on the model's classification accuracy. If the student's answer contains an error, the model proceeds to identify the specific location of the error for the EL task. To assess localization precision, we use the critic model QwQ-32B as a Judge due to its superior performance in error localization (Zheng et al., 2024). When no error location is identified (None), we cross-check with the ground truth of error detection—if the student's answer is correct, the localization is deemed correct; otherwise, it is marked as wrong localization. The results are reported based on a single run for consistency. Further details on datasets (e.g., AIHub, Fermat) and evaluation protocols are provided in the Appendix.

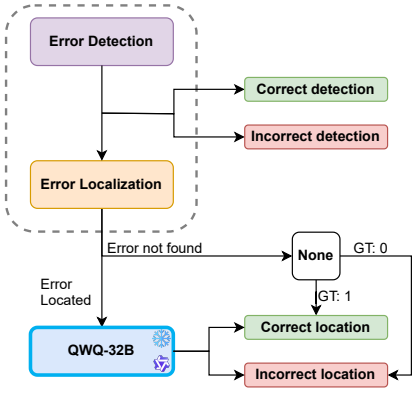

Figure 5: VLM evaluation procedure for student solutions, encompassing Error Detection (ED) and Error Localization (EL). ED serves to identify the presence of errors, while EL is responsible for determining the specific location of any detected error.

Experimental results The experiments are conducted on two tasks–Error Detection and Error Localization, across the AIHub and FERMAT datasets. Table 1 compares our VEHME-Qwen2.5-VL-7B against four open-source baselines and four closed-source systems on the AIHub and FERMAT benchmarks. Among the open-source group, our model achieves a clear lead in both Error Detection (ED) and Error Localization (EL). On AIHub, VEHME records 73.01% ED accuracy (49.22% F1) and 61.13% EL accuracy (58.18% F1), substantially outperforming Qwen2.5-VL-7B-Instruct (46.68%Acc/ 31.48% F1 for ED, 38.00% Acc/ 33.75% F1 for EL), Pixtral-12B (52.67% / 31.69% ED, 32.20% / 38.35% EL), Phi-4-multimodal-instruct, and Llama-3.2-11B. This margin highlights the effectiveness of our data synthesis pipeline and dual-phase training in refining both expression recognition and precise localization.

Models	AIHub				FERMAT			
	ED		EL		ED		EL	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Closed-source Models								
GPT-4o	73.00	48.67	62.88	61.01	66.62	**58.57**	55.04	66.07
GPT-4o-mini	57.15	32.97	38.98	49.18	**81.75**	56.62	47.18	62.61
Gemini-2.0-Flash	**75.40**	**50.30**	**67.40**	**65.73**	75.22	43.13	**64.24**	**75.13**
Gemini-2.5-Flash-Preview	71.01	46.91	63.00	65.68	80.12	45.68	63.35	74.72
Open-source Models								
Phi-4-multimodal-instruct	38.57	29.39	35.93	21.33	36.20	22.33	16.91	15.15
Qwen2.5-VL-7B-Instruct	46.68	31.48	38.00	33.75	44.81	26.04	24.33	32.18
Pixtral-12B	52.67	31.69	32.20	38.35	41.39	25.56	10.53	3.52
Llama-3.2-11B-Vision-Instruct	19.49	18.58	44.78	5.62	32.49	22.15	21.07	22.67
VEHME-Qwen2.5-VL-7B	**73.01**	**49.22**	**61.13**	**58.18**	**62.61**	**29.81**	**31.90**	**44.36**

Table 1: Performance comparisons of state-of-the-art Vision-Language Models on two datasets: AIHub and FERMAT (Nath et al., 2025). The evaluation metrics include Accuracy (Acc) and F1 score (F1). **ED**: Error Detection, **EL**: Error Localization. All of the reported results are in percentages (%).

On FERMAT, open-source models generally struggle, but our approach again leads the pack with 62.61% ED accuracy (29.81% F1) and 31.90% EL accuracy (44.36% F1). Notably, Pixtral-12B exhibits an unusual split—high ED accuracy (41.39%) yet very low EL performance (10.53% Acc/3.52% F1)–which echoes the observations in the original FERMAT study: its format alignment and output consistency are suboptimal for localization tasks (Nath et al., 2025). In contrast, our model's balanced gains across both metrics demonstrate robust adherence to the expected answer format and reliable spatial grounding.

When we turn to closed-source systems, Gemini-2.0-Flash remains the top performer overall (75.40%/50.30% ED, 67.40%/65.73% EL on AIHub; 75.22%/43.13% ED, 64.24%/75.13% EL on FERMAT), followed closely by GPT-4o and its mini variant. These proprietary models, however, exceed tens or even hundreds of billions of parameters, whereas our Qwen2.5-VL-7B backbone contains only 7 billion parameters. The result demonstrates that our lightweight model surpasses other open-source approaches and nearly matches closed-source models, while operating at a fraction of the scale of closed systems, underscoring the efficiency gains enabled by our targeted data generation and training regimen. The qualitative result and analysis are presented in Appendix A.7. Addtionally, we provide analysis of experimental result across different educational levels in Appendix A.5.

Models	ED		EL	
	Acc	F1	Acc	F1
Ours	**73.01**	**49.22**	61.13	**58.18**
-EVPM	71.98	48.87	**61.70**	56.59
-SFT	63.24	41.47	52.67	33.41
-RL	46.90	33.82	39.24	29.59
Baseline	46.68	31.48	38.00	33.75

Table 2: Ablation study results validating our proposed components on the AIHub dataset.

4.4 Ablation Study

Component Ablation In this section, we perform a comprehensive ablation study on the balanced AIHub dataset to quantify the impact of each major component in our architecture. Table 2 reports performance when ablating specific modules from our full model. Omitting the bounding-box visual hints (–EVPM) yields a modest decrease in ED accuracy (73.01%→71.98%) and a larger decline in EL F1 (58.18%→56.59%), underscoring the necessity of spatial grounding. Skipping supervised fine-tuning (–SFT) substantially degrades both ED and EL, indicating that task-specific supervision is crucial for aligning the vision–language backbone with the reasoning steps and expected output format. The removal of reinforcement learning (–RL) results in a dramatic collapse, particularly in EL F1 (58.18%→29.59%), highlighting RL's indispensable role in honing expression comprehension and precise localization. Our intact model consistently outperforms all ablated variants across every metric in both tasks.

Models	ED		EL	
	Acc	F1	Acc	F1
GPT-4o	73.00	72.93	63.00	62.36
GPT-4o-mini	48.00	39.22	33.00	29.26
Gemini-2.0-Flash	71.00	71.00	**65.00**	**64.83**
Gemini-2.5-Flash*	**76.00**	**74.78**	59.00	58.80
Phi-4*	33.00	25.14	36.00	33.33
Qwen2.5-VL*	50.00	35.87	46.00	45.45
Pixtral*	41.00	25.38	29.00	28.82
Llama-3.2*	21.00	18.93	48.00	34.04
Ours	**75.00**	**50.24**	**66.00**	**65.32**
-EVPM	62.00	42.26	62.00	61.00
-SFT	61.00	39.02	58.00	53.21
-RL	43.00	31.69	44.00	41.67

Table 3: Ablation study results validating our proposed components on the "*Challenging Subset of AIHub*" dataset. *We use the same models as in the main experiment; we omit the suffix for brevity.

Ablation Under Heavy Rotations In our analysis, most of the standard dataset has mild rotation ($< 15°$), which might explain the relatively small global improvement from EVPM. To better understand EVPM's contribution, we constructed a "*Challenging Subset of AIHub*" by identifying the 100 handwritten data points with the most heavily rotated mathematical expressions (top-100; Mean: $21.81°$). Examples of this subset can be found in Appendix A.6. Table 3 reports performance when the ablation study is conducted in the challenging subset. These results demonstrate that EVPM makes a significant contribution under heavy rotations. On this subset, removing EVPM causes a substantial drop in performance (ED: $75\% \rightarrow 62\%$, and EL: $66\% \rightarrow 62$), even with all other training and inference settings held constant. Compared with the closed-source models, our approach significantly narrows the gap in error detection and even outperforms them in error localization. Specifically, for ED, our method (75%/50.24%) comes close to the strongest closed-source baselines such as GPT-4o (73%/72.93%) and Gemini-2.5-Flash-Preview (76%/74.78%), showing that our system can achieve comparable detection accuracy under challenging rotations. More notably, in EL, our approach achieves 66% Acc/65.32% F1, surpassing both GPT-4o (63%/62.36%) and Gemini-2.5-Flash-Preview (59%/58.80%). This indicates that while the closed-source models remain competitive in ED, our method demonstrates a clear advantage in localizing errors under heavy rotational noise.

Image Resolutions	ED		EL	
	Acc	F1	Acc	F1
224×224	73.01	49.22	61.13	**58.18**
448×448	**75.66**	**50.42**	63.87	56.57
768×768	75.29	50.11	**64.58**	56.50

Table 4: Ablation study results with different image resolutions on the AIHub dataset.

Image Resolution Ablation In our main experiment, we resize the input images down to 224x224 for computational efficiency. To assess the impact of image resolution on performance, we conduct an ablation study using images with varying resolutions. Table 4 shows VEHME's performance across resolutions. Increasing the resolution generally improves both error detection and localization accuracies, with the largest gains observed when moving from 224x224 to 448x448. This improvement is likely due to better preservation of fine-grained visual details in handwritten expressions. However, further increasing the resolution to 768x768 yields only marginal improvements, with performance comparable to that at 448x448. These results suggest that while higher resolutions can capture more detail, the benefits plateau beyond a certain point, making the additional computational cost less justifiable.

5 Conclusion

We present VEHME, a novel Vision-Language Model tailored for evaluating open-form handwritten mathematical expressions with high accuracy, interpretability, and scalability. Leveraging a dual-phase training strategy, supervised fine-tuning on expert reasoning traces and reinforcement learning via GRPO, VEHME delivers fine-grained error detection and localization across visually diverse student responses. The Expression-Aware Visual Prompting Module further enhances the model's spatial reasoning capabilities. Experiments on the AIHub and FERMAT datasets show that VEHME consistently outperforms existing open-source models and achieves performance competitive with state-of-the-art proprietary systems. These findings underscore VEHME's promise as an effective, open-access solution for scalable, automated assessment in educational settings.

Limitations

While VEHME demonstrates promising results in evaluating handwritten mathematical expressions,

several limitations should be acknowledged. First, due to data-sharing restrictions from the source provider (AIHub), the original math problems and student responses cannot be publicly released outside Korea, limiting reproducibility. Secondly, while VEHME is robust to varied inputs, it may struggle with extremely illegible handwriting, low-quality images, or unconventional notation.

Thirdly, QwQ-32B may propagate biases in our evaluation pipeline, stemming from length-based tendencies or anchoring effects introduced during training and reward design. While output length bias was explicitly controlled, addressing deeper preference and anchoring biases remains an open challenge for future work.

Lastly, the reinforcement learning setup, though effective, can be sensitive to reward design and may introduce training instability. Future work will explore more accessible datasets, improved robustness, and interpretable learning signals.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2024–00456247, No. RS-2023–00218913, No. RS-2023-00219959) and by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No. RS-2025-25443718, Next-HCAI Project, No.RS-2022- II220608/2022-0-00608, Artificial intelligence research about multimodal interactions for empathetic conversations with humans, No.IITP-2025-RS-2024-00360227, Leading Generative AI Human Resources Development, No.RS-2025-25442824, AI Star Fellowship Program(Ulsan National Institute of Science and Technology), No.RS-2020-II201336, Artificial Intelligence graduate school support(UNIST) and No.RS-2019-II191906, Artificial Intelligence Graduate School Program(POSTECH).

This work utilized datasets provided by the Open AI Dataset Project (AIHub, South Korea), which are publicly available through AIHub (www.aihub.or.kr).

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *Preprint*, arXiv:2308.12966.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, and 1 others. 2025. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*.

Sami Baral, Li Lucy, Ryan Knight, Alice Ng, Luca Soldaini, Neil Heffernan, and Kyle Lo. 2025. Drawedumath: Evaluating vision language models with expert-annotated students' hand-drawn math images. In *Proceedings of the 2025 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Mary E Callahan, Emily B Brant, Deepika Mohan, Marie K Norman, Robert M Arnold, and Douglas B White. 2021. Leveraging technology to overcome the "scalability problem" in communication skills training courses. *ATS scholar*, 2(3):327–340.

Kam-Fai Chan and Dit-Yan Yeung. 1998. Elastic structural matching for online handwritten alphanumeric character recognition. In *Proceedings. Fourteenth International Conference on Pattern Recognition (Cat. No. 98EX170)*, volume 2, pages 1508–1511. IEEE.

Kam-Fai Chan and Dit-Yan Yeung. 2000. Mathematical expression recognition: a survey. *International Journal on Document Analysis and Recognition*, 3:3–15.

Ekawat Chaowicharat and Natasha Dejdumrong. 2023. A step toward an automatic handwritten homework grading system for mathematics. *Information Technology and Control*, 52(1):169–184.

Dongping Chen, Ruoxi Chen, Shilin Zhang, Yaochen Wang, Yinuo Liu, Huichi Zhou, Qihui Zhang, Yao Wan, Pan Zhou, and Lichao Sun. 2024a. Mllm-as-a-judge: Assessing multimodal llm-as-a-judge with vision-language benchmark. In *Forty-first International Conference on Machine Learning*.

Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. 2024b. Alphamath almost zero: Process supervision without process. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Hardy Chen, Haoqin Tu, Fali Wang, Hui Liu, Xianfeng Tang, Xinya Du, Yuyin Zhou, and Cihang Xie. 2025a. Sft or rl? an early investigation into training r1-like reasoning large vision-language models. *arXiv preprint arXiv:2504.11468*.

Nuo Chen, Zhiyuan Hu, Qingyun Zou, Jiaying Wu, Qian Wang, Bryan Hooi, and Bingsheng He. 2025b. Judgelrm: Large reasoning models as a judge. *ArXiv*, abs/2504.00050.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, and 1 others. 2024c. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 24185–24198.

Yihe Deng, Hritik Bansal, Fan Yin, Nanyun Peng, Wei Wang, and Kai-Wei Chang. 2025. Openvlthinker: An early exploration to complex vision-language reasoning via iterative self-improvement. *ArXiv*, abs/2503.17352.

Yuntian Deng, Anssi Kanervisto, Jeffrey Ling, and Alexander M Rush. 2017. Image-to-markup generation with coarse-to-fine attention. In *International Conference on Machine Learning*, pages 980–989. PMLR.

Saibo Geng, Martin Josifoski, Maxime Peyrard, and Robert West. 2023. Grammar-constrained decoding for structured NLP tasks without finetuning. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Ramya S Gowda and V Suma. 2017. A comparative analysis of traditional education system vs. e-learning. In *2017 International conference on innovative mechanisms for industry applications (ICIMIA)*, pages 567–571. IEEE.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, and 1 others. 2024. A survey on llm-as-a-judge. *arXiv preprint arXiv:2411.15594*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025a. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Hong-Yu Guo, Chuang Wang, Fei Yin, Xiao-Hui Li, and Cheng-Lin Liu. 2025b. Vision–language pre-training for graph-based handwritten mathematical expression recognition. *Pattern Recognition*, 162:111346.

Hong-Yu Guo, Fei Yin, Jian Xu, and Cheng-Lin Liu. 2025c. Hie-vl: A large vision-language model with hierarchical adapter for handwritten mathematical expression recognition. In *ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

Jujie He, Tianwen Wei, Rui Yan, Jiacai Liu, Chaojie Wang, Yimeng Gan, Shiwen Tu, Chris Yuhao Liu, Liang Zeng, Xiaokun Wang, Boyang Wang, Yongcong Li, Fuxiang Zhang, Jiacheng Xu, Bo An, Yang Liu, and Yahui Zhou. 2024. Skywork-o1 open series. `https://huggingface.co/Skywork`.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Hyoungwook Jin, Yoonsu Kim, Dongyun Jung, Seungju Kim, Kiyoon Choi, Jinho Son, and Juho Kim. 2025. Investigating large language models in diagnosing students' cognitive skills in math problem-solving. *arXiv preprint arXiv:2504.00843*.

Glenn Jocher, Jing Qiu, and Ayush Chaurasia. 2023. Ultralytics yolo (version 8.0.0) [computer software].

Muhammad Khalifa, Rishabh Agarwal, Lajanugen Logeswaran, Jaekyeom Kim, Hao Peng, Moontae Lee, Honglak Lee, and Lu Wang. 2025. Process reward models that think. *arXiv preprint arXiv:2504.16828*.

Gerd Kortemeyer, Julian Nöhl, and Daria Onishchuk. 2024. Grading assistance for a handwritten thermodynamics exam using artificial intelligence: An exploratory study. *Physical Review Physics Education Research*, 20(2):020144.

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

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, and 1 others. 2024. Rewardbench: Evaluating reward models for language modeling. *arXiv preprint arXiv:2403.13787*.

R Dwight Laws, Scott L Howell, and Nathan K Lindsay. 2003. Scalability in distance education:"can we have our cake and eat it too?". *Online Journal of Distance Learning Administration*, 6(4):80–89.

Bohan Li, Ye Yuan, Dingkang Liang, Xiao Liu, Zhilong Ji, Jinfeng Bai, Wenyu Liu, and Xiang Bai. 2022. When counting meets hmer: counting-aware network for handwritten mathematical expression recognition. In *European conference on computer vision*, pages 197–214. Springer.

Feng Li, Qing Jiang, Hao Zhang, Tianhe Ren, Shilong Liu, Xueyan Zou, Huaizhe Xu, Hongyang Li, Jianwei Yang, Chunyuan Li, Lei Zhang, and Jianfeng Gao. 2024a. Visual in-context prompting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12861–12871.

Junsong Li, Jie Zhou, Yutao Yang, Bihao Zhan, Qianjun Pan, Yuyang Ding, Qin Chen, Jiang Bo, Xin Lin, and Liang He. 2025. Teaching llms for step-level automatic math correction via reinforcement learning. *arXiv preprint arXiv:2503.18432*.

Xiaoshuo Li, Tiezhu Yue, Xuanping Huang, Zhe Yang, and Gang Xu. 2019. Bags: An automatic homework grading system using the pictures taken by smart phones. *arXiv preprint arXiv:1906.03767*.

Zhe Li, Wentao Yang, Hengnian Qi, Lianwen Jin, Yichao Huang, and Kai Ding. 2024b. A tree-based model with branch parallel decoding for handwritten mathematical expression recognition. *Pattern Recognition*, 149:110220.

John JH Lin. 2024. Ai-assisted evaluation of problem-solving performance using eye movement and handwriting. *Journal of Research on Technology in Education*, pages 1–25.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916.

Ziyu Liu, Zeyi Sun, Yuhang Zang, Xiao wen Dong, Yuhang Cao, Haodong Duan, Dahua Lin, and Jiaqi Wang. 2025. Visual-rft: Visual reinforcement fine-tuning. *ArXiv*, abs/2503.01785.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Ryan Mok, Faraaz Akhtar, Louis Clare, Christine Li, Jun Ida, Lewis Ross, and Mario Campanelli. 2024. Using ai large language models for grading in education: A hands-on test for physics. *arXiv preprint arXiv:2411.13685*.

Ryosuke Nakamoto, Brendan Flanagan, Yiling Dai, Taisei Yamauchi, Kyosuke Takami, and Hiroaki Ogata. 2023. Enhancing self-explanation learning through a real-time feedback system: An empirical evaluation study. *Sustainability*, 15(21):15577.

Oikantik Nath, Hanani Bathina, Mohammed Safi Ur Rahman Khan, and Mitesh M Khapra. 2025. Can vision-language models evaluate handwritten math? In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14784–14814, Vienna, Austria. Association for Computational Linguistics.

OpenAI. 2024. Learning to reason with llms. Accessed: 2025-05-19.

QwenTeam. 2025. Qwq-32b: Embracing the power of reinforcement learning.

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

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Haozhan Shen, Zilun Zhang, Qianqian Zhang, Ruochen Xu, and Tiancheng Zhao. 2025. Vlm-r1: A stable and generalizable r1-style large vision-language model. 2025. *URL https://arxiv. org/abs/2504.07615*.

Luong Trung, Xinbo Zhang, Zhanming Jie, Peng Sun, Xiaoran Jin, and Hang Li. 2024. Reft: Reasoning with reinforced fine-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7601–7614.

Thanh-Nghia Truong, Cuong Tuan Nguyen, and Masaki Nakagawa. 2022. Syntactic data generation for handwritten mathematical expression recognition. *Pattern Recognition Letters*, 153:83–91.

Benjamin Wagstaff, Chiao Lu, and Xiang'Anthony' Chen. 2019. Automatic exam grading by a mobile camera: Snap a picture to grade your tests. In *Companion Proceedings of the 24th International Conference on Intelligent User Interfaces*, pages 3–4.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *ACL (1)*, pages 9426–9439.

Weiyun Wang, Zhangwei Gao, Lianjie Chen, Zhe Chen, Jinguo Zhu, Xiangyu Zhao, Yangzhou Liu, Yue Cao, Shenglong Ye, Xizhou Zhu, and 1 others. 2025. Visualprm: An effective process reward model for multimodal reasoning. *arXiv preprint arXiv:2503.10291*.

Changjie Wu, Jun Du, Yunqing Li, Jianshu Zhang, Chen Yang, Bo Ren, and Yiqing Hu. 2022. Tdv2: A novel tree-structured decoder for offline mathematical expression recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 2694–2702.

Jin-Wen Wu, Fei Yin, Yan-Ming Zhang, Xu-Yao Zhang, and Cheng-Lin Liu. 2020. Handwritten mathematical expression recognition via paired adversarial learning. *International Journal of Computer Vision*, 128:2386–2401.

Jin-Wen Wu, Fei Yin, Yan-Ming Zhang, Xu-Yao Zhang, and Cheng-Lin Liu. 2021. Graph-to-graph: towards accurate and interpretable online handwritten mathematical expression recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 2925–2933.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang,

Jingren Zhou, Junyang Lin, Kai Dang, and 23 others. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Wentao Yang, Zhe Li, Dezhi Peng, Lianwen Jin, Mengchao He, and Cong Yao. 2023. Read ten lines at one glance: line-aware semi-autoregressive transformer for multi-line handwritten mathematical expression recognition. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 2066–2077.

Edward Yeo, Yuxuan Tong, Xinyao Niu, Graham Neubig, and Xiang Yue. 2025. Demystifying long chain-of-thought reasoning in LLMs. In *ICLR 2025 Workshop on Navigating and Addressing Data Problems for Foundation Models*.

Kechi Zhang, Ge Li, Yihong Dong, Jingjing Xu, Jun Zhang, Jing Su, Yongfei Liu, and Zhi Jin. 2024a. Codedpo: Aligning code models with self generated and verified source code. *arXiv preprint arXiv:2410.05605*.

Lingyu Zhang, Yafeng Yin, Lei Xie, and Sanglu Lu. 2020. Hmwkcheck: a homework auto-checking system based on arithmetic operation recognition using smartphones. In *Adjunct Proceedings of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers*, UbiComp/ISWC '20 Adjunct, page 172–175, New York, NY, USA. Association for Computing Machinery.

Mengxue Zhang, Zichao Wang, Richard Baraniuk, and Andrew Lan. 2021. Math operation embeddings for open-ended solution analysis and feedback. *arXiv preprint arXiv:2104.12047*.

Tianyang Zhang, Zhuoxuan Jiang, Haotian Zhang, Lin Lin, and Shaohua Zhang. 2025. Mathmistake checker: A comprehensive demonstration for step-by-step math problem mistake finding by prompt-guided llms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 29730–29732.

Yuxiang Zhang, Shangxi Wu, Yuqi Yang, Jiangming Shu, Jinlin Xiao, Chao Kong, and Jitao Sang. 2024b. o1-coder: an o1 replication for coding. *arXiv preprint arXiv:2412.00154*.

Wenqi Zhao and Liangcai Gao. 2022. Comer: Modeling coverage for transformer-based handwritten mathematical expression recognition. In *European conference on computer vision*, pages 392–408. Springer.

Yuze Zhao, Jintao Huang, Jinghan Hu, Xingjun Wang, Yunlin Mao, Daoze Zhang, Zeyinzi Jiang, Zhikai Wu, Baole Ai, Ang Wang, Wenmeng Zhou, and Yingda Chen. 2024. Swift:a scalable lightweight infrastructure for fine-tuning. *Preprint*, arXiv:2408.05517.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2024. Processbench: Identifying process errors in mathematical reasoning. *arXiv preprint arXiv:2412.06559*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

Hengguang Zhou, Xirui Li, Ruochen Wang, Minhao Cheng, Tianyi Zhou, and Cho-Jui Hsieh. 2025. R1-zero's "aha moment" in visual reasoning on a 2b non-sft model. *ArXiv*, abs/2503.05132.

A Appendix

A.1 Datasets

Figure 6 shows representative samples from the AIHub dataset and the FERMAT dataset. Both contain math problems paired with student solutions, but they differ in format and collection style.

AIHub comprises 183,085 handwritten student responses corresponding to 30,050 unique problem questions, originally authored in Korean. To mitigate potential performance degradation due to cross-lingual inputs, we filtered out student answers containing non-English characters. Given that many questions include figures, after translating to English, we further rendered the translated content into LaTeX and converted it into images to ensure compatibility with visual input processing.

As shown in Figure 7, the original AIHub dataset exhibits a significant class imbalance, with approximately 72% of answers being correct and 28% incorrect. To mitigate this skew, we curate the training data as follows: **Step 1:** We discard problems that had only correct or only incorrect student answers, leaving approximately 150,000 problem-answer pairs. **Step 2:** For each question, we sample an equal number of correct and incorrect answers to form a balanced training set. In the end, we obtained a balanced dataset of correct and incorrect solutions from students with 81,394 training samples and 9,062 test samples.

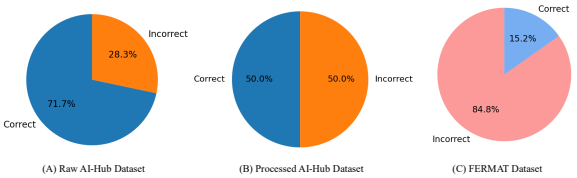

Figure 7: Dataset distribution for correct/incorrect answer.

FERMAT provides a complementary evaluation focused on error analysis, so incorrect solutions significantly outnumber correct ones at an approximate ratio of 85:15, which is different from the balanced distribution in the filtered AIHub dataset. It contains 2,244 manually curated solutions across eight mathematical domains from arithmetic to calculus. Each entry contains: the original question, a handwritten solution image with intentional er-

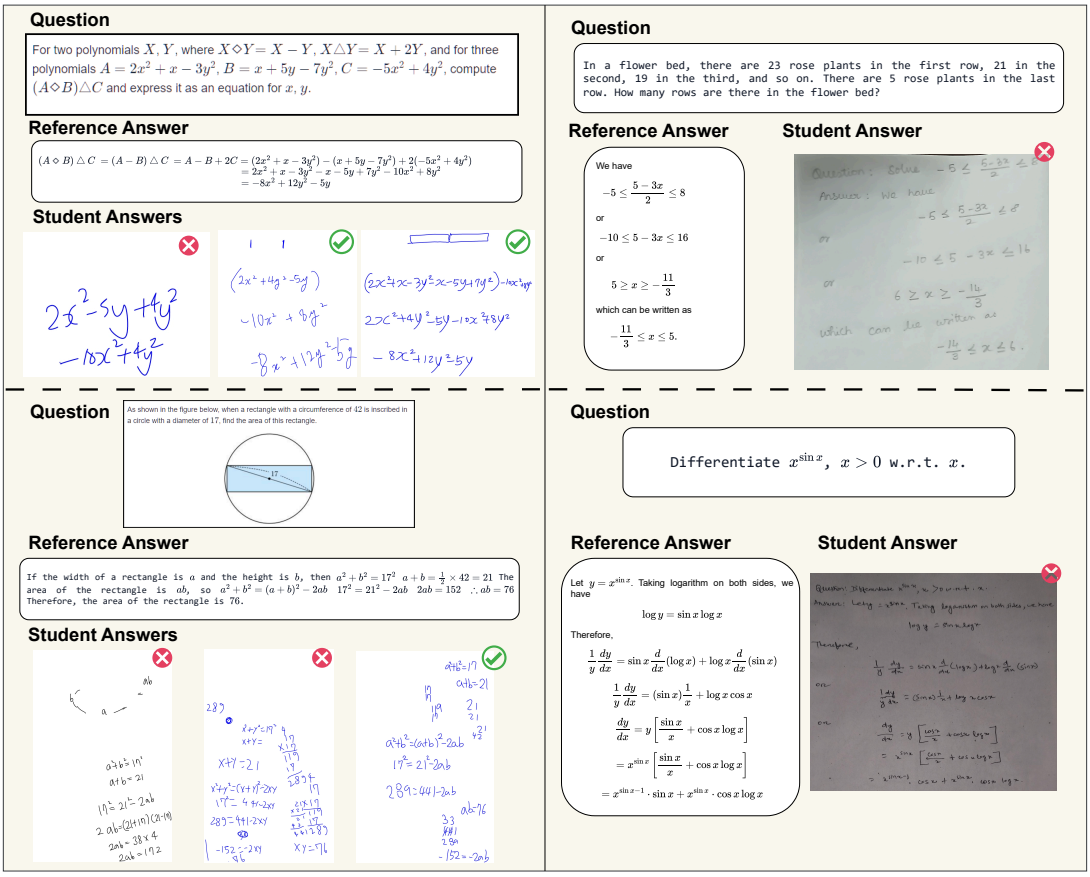

Figure 6: Data samples of AIHub (left) and FERMAT (Right) Datasets.

rors, the gold-standard correct answer, and error detection labels. Due to its small size, we opt not to perform any additional filtering, as doing so could lead to an insufficient amount of data for fine-tuning. We split the dataset into 70% training and 30% test sets.

A.2 Data synthesis from QwQ-32B for SFT

Figure 8 illustrates some data synthesized from QwQ-32B, showcasing its reasoning process for assessing correctness and localizing errors in students' handwritten responses.

Please refer to Algorithm 1 for a detailed view of the data synthesis procedure.

The grammar G used in the decoding process is as follows:

$$
\begin{aligned}
\textit{root} &\rightarrow C \text{ “</correctness>” } L \\
C &\rightarrow \text{“correct”} \mid \text{“incorrect”} \\
L &\rightarrow \text{“<localization>” } E \text{ “</localization>”} \\
E &\rightarrow \text{“None”} \mid \Sigma_{\text{math}}{}^{*} \\
&\quad \mid \text{“Lack of intermediate steps”}
\end{aligned}
$$

where Σ_{math} is the set of valid characters repre-

Algorithm 1 Post-Processing for Generated Data

Input: Output text T, Prompt P, Maximum token limit M, Grammar constraint G, Language model LLM

Output: Final response with correct format

if TokenCount(T) < M **then**
  return T
end if
$T' \leftarrow$ TruncateLastThought(T)
$T'' \leftarrow T' \,||\, $ "</think>\n\n<correctness>"
$P' \leftarrow P \,||\, T''$
Configure LLM with grammar constraint G
$R \leftarrow$ LLM.Generate(P')
return $P' \,||\, R$

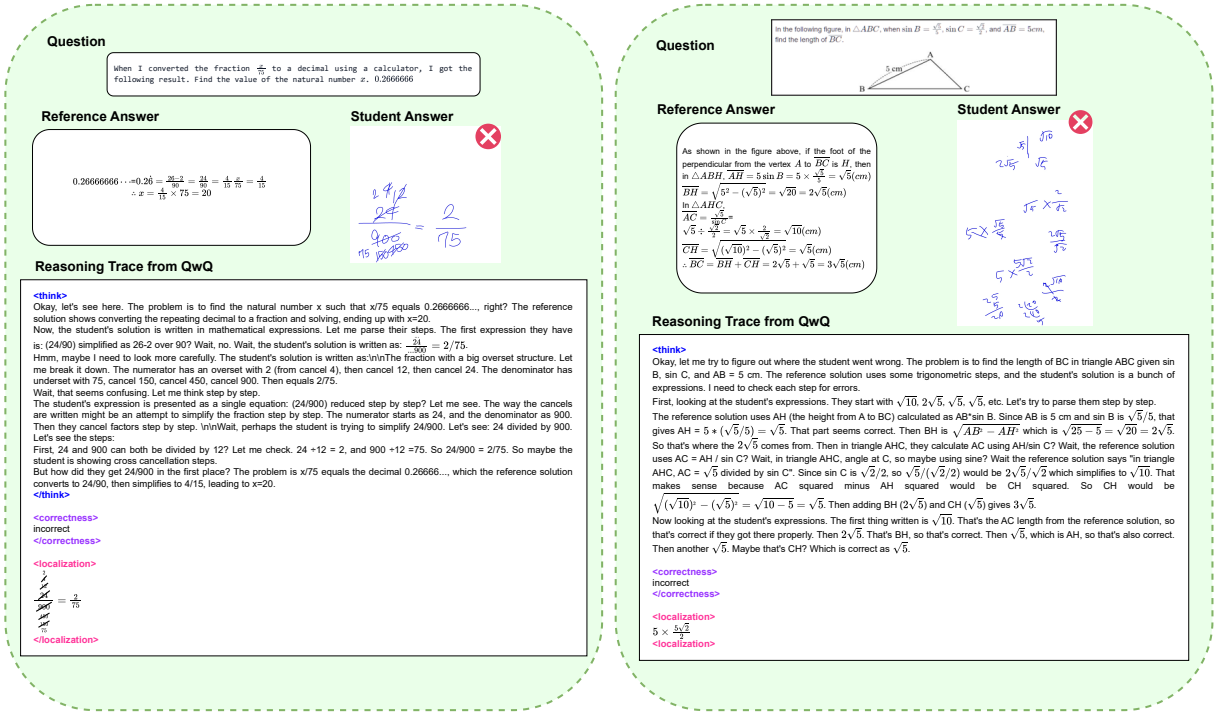

Figure 8: Examples of data synthesized from QwQ-32B for SFT. The model inputs—questions, reference answers, and student responses—are provided in LaTeX format. The figure shown here is for illustrative purposes only; no images are given to the model.

senting mathematical expressions

Σ_{math} = [a-zA-Z0-9\{}_^$\().,;'"-=<>+|*/]*

A.3 Expression-aware Visual Prompting

Please refer to Algorithm 2 for a formal view of how we synthesize multi-line mathematical expressions.

Algorithm 2 Multi-Line Expression Canvas Synthesis

Input: Set of HMES files $\mathcal{H}$, padding bound P, rotation bound θ

Output: Synthesized canvas $\mathcal{C}$ and list of oriented bounding boxes $\mathcal{B}$

Render each expression $h_i \in \mathcal{H}$ to image I_i
Compute canvas size from maximum image width and height
Initialize blank canvas $\mathcal{C}$ and vertical offset $y \leftarrow 0$
for all images I_i **do**
    Sample rotation angle $\alpha \sim \mathcal{U}(-\theta, \theta)$
    Rotate image with α to obtain I_i^{rot}
    Sample horizontal offset

$$x \sim \mathcal{U}\left(0, W - \text{width}\left(I_i^{\text{rot}}\right)\right)$$

    Compute rotated bounding box corners around image center
    Add (x, y)-shifted rotated corners to $\mathcal{B}$
    Paste I_i^{rot} onto canvas $\mathcal{C}$ at (x, y)
    Update $y \leftarrow y + h_i^{\text{rot}} + \text{randint}(-P, P)$
end for
return Cropped canvas $\mathcal{C}$ and bounding box list $\mathcal{B}$

A.4 Implementation Details

To train our model, we apply LoRA (Hu et al., 2022) to all MLP layers with a rank of 8 and an α-scaling factor of 32. Optimization is performed using the AdamW optimizer (Loshchilov and Hutter, 2017) with a learning rate of 5e-5, cosine learning rate decay, and a warm-up ratio of 0.05. We set a global batch size of 192 and apply a weight decay of 0.1 to mitigate overfitting. Training is conducted over a single epoch, requiring approximately 6 GPU-days on NVIDIA A100s.

To train our EVPM component, we utilize Yolov11 (Jocher et al., 2023) as the backbone of the bounding box detector. We synthesize $10,000$ training data, and an additional 1000 validation data. Optimization is performed using the AdamW optimizer (Loshchilov and Hutter, 2017) with a learning rate of 0.01. We set a global batch size of 16. Training the EVPM is conducted over 200 epochs, requiring only a few hours on a single NVIDIA RTX 3090.

To ensure efficiency during both supervised and reinforcement learning stages, we adopt the `ms-swift` framework (Zhao et al., 2024), which provides optimized memory utilization and high-throughput training for vision-language models. For inference, we employ the `vllm` framework (Kwon et al., 2023), enabling fast, memory-efficient evaluation across all open-source models, including our own. The prompts for inference are presented in Figure 9 and Figure 10.

A.5 Quantitative Experiment on Different Education Levels

Educational datasets naturally vary in complexity depending on the target grade level: primary school problems tend to be more straightforward and involve simpler arithmetic or reasoning steps, whereas middle and high school problems increasingly incorporate multi-step reasoning, algebraic manipulation, and abstract logic. Since error detection (ED) and error localization (EL) are reasoning-intensive tasks, it is important to evaluate whether models maintain consistent performance across different difficulty levels. Table 5 reports results on the AIHub dataset split by education levels (Primary, Middle, and High School). The evaluation again covers Error Detection (ED) and Error Localization (EL).

Gemini-2.0-Flash emerges as the strongest proprietary baseline across all education levels. It records peak accuracy at the High School level (76.40% ED / 68.96% EL), showing consistent gains as the problem difficulty increases. GPT-4o also performs robustly, particularly on Middle School data (75.24% ED / 65.71% EL). The open-source VLMs generally struggle to match closed-source performance, with accuracies ranging between 18–53%. Pixtral-12B achieves stable ED accuracy ($\approx 52\%$) across levels but remains weak in EL ($\approx 32\%$). Llama-3.2-11B shows relatively higher EL than other open-source peers ($\approx 45\%$), though at the cost of very poor ED performance ($\approx 19\%$). Phi-4 and Qwen2.5-VL-7B-Instruct sit in the middle range, with incremental improvements from Primary to High School, but failing to scale effectively. VEHME–Qwen2.5-VL-7B consistently outperforms other open-source competitors across

Sytem Prompt

You are a helpful assistant. You first think about the reasoning process in the mind and then provide the user with the answer.

User Prompt

The following contains a math problem consisting of both a Question and a Student Answer. Your task is to analyze the Student Answer to determine whether there is any error. Begin by providing a detailed reasoning for your analysis, explaining where and why you believe an error is present or absent in the Student Answer. After the reasoning, provide a final output indicating whether the Student Answer is correct or incorrect. Please follow the exact format below without adding any extra information:

<think>
Detailed reasoning for your analysis, explaining where and why you believe an error is present or absent in the Student Answer.
</think>

<correctness>
Correct or Incorrect
</correctness>

<localization>
The localization of the error in the Student Answer. Write None if there is no error.
</localization>

Few-shot Prompt

<think>
The Student Answer correctly identifies the slope and y-intercept of the original line as 2 and 3, respectively. The equation of the translated line is also correctly written as \\(y = 2x + 3 + a \\). The substitution of the point (5, 6) into the equation is also done correctly: \\(6 = 2 \\times 5 + 3 + a \\), which simplifies to \\(6 = 13 + a \\). The final step of solving for \\(a \\) is correct as well: \\(a = -7 \\). The Student Answer follows the same logical steps as the Reference Answer, and no errors are present in the calculations or reasoning.
</think>

<correctness>
Correct
</correctness>

<localization>
None
</localization>

Input

Question: <image>
Reference answer: reference answer
Student answer : <image>

Figure 9: Prompt examples for the inferencing on AIHUB.

Sytem Prompt

You are a helpful assistant. You first think about the reasoning process in the mind and then provide the user with the answer.

User Prompt

The following is a single image containing both a Question and a Student Answer. Your task is to analyze the Student Answer to determine whether there is any error. Begin by providing a detailed reasoning for your analysis, explaining where and why you believe an error is present or absent in the Student Answer. After the reasoning, provide a final output indicating whether the Student Answer is correct or incorrect. The error location must be a specific mathematical expression in the Student Answer. Please follow the exact format below without adding any extra information:

<think>
Detailed reasoning for your analysis, explaining where and why you believe an error is present or absent in the Student Answer.
</think>

<correctness>
Correct or Incorrect
</correctness>

<localization>
The localization of the error in the Student Answer. Write None if there is no error.
</localization>

Few-shot Prompt

<think>
The Student Answer correctly identifies the slope and y-intercept of the original line as 2 and 3, respectively. The equation of the translated line is also correctly written as \\(y = 2x + 3 + a \\). The substitution of the point (5, 6) into the equation is also done correctly: \\(6 = 2 \\times 5 + 3 + a \\), which simplifies to \\(6 = 13 + a \\). The final step of solving for \\(a \\) is correct as well: \\(a = -7 \\). The Student Answer follows the same logical steps as the Reference Answer, and no errors are present in the calculations or reasoning.
</think>

<correctness>
Correct
</correctness>

<localization>
None
</localization>

Input

Question and Student answer: <image>
Reference answer: reference answer

Figure 10: Prompt examples for the inferencing on FERMAT.

Models	AIHub – Primary School				AIHub – Middle School				AIHub – High School			
	ED		EL		ED		EL		ED		EL	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Closed-source Models												
GPT-4o	74.00	49.35	64.81	64.51	**75.24**	**75.17**	65.71	65.68	75.34	**75.34**	66.25	66.04
GPT-4o-mini	58.60	34.83	40.45	38.65	56.10	31.45	37.42	34.34	55.90	31.52	39.65	35.65
Gemini-2.0-Flash	**75.59**	**50.43**	**67.58**	**67.52**	74.98	50.01	**66.84**	**66.76**	**76.40**	50.95	**68.96**	**68.85**
Gemini-2.5-Flash-Preview	71.20	46.97	62.94	62.67	70.91	46.86	62.56	62.35	70.70	70.23	64.99	64.82
Open-source Models												
Phi-4-multimodal-instruct	38.01	28.91	35.27	32.79	39.84	30.23	36.50	34.32	35.59	27.44	36.17	34.13
Qwen2.5-VL-7B-Instruct	46.66	32.58	39.01	38.69	47.53	33.35	39.33	39.26	48.65	33.65	37.72	37.71
Pixtral-12B	52.70	31.90	32.03	31.52	52.76	31.41	32.19	31.31	52.71	32.14	32.88	32.25
Llama-3.2-11B-Vision-Instruct	19.40	18.34	44.58	33.29	19.77	18.78	44.55	33.40	18.67	18.15	46.42	32.77
VEHME-Qwen2.5-VL-7B	**72.10**	**48.60**	**60.04**	**59.85**	**73.77**	**49.76**	**61.50**	**61.33**	**73.40**	**49.35**	**63.83**	**63.45**

Table 5: Performance comparisons of state-of-the-art Vision-Language Models on different education levels in the AIHub dataset. The evaluation metrics include Accuracy (Acc) and F1 score (F1). **ED**: Error Detection, **EL**: Error Localization. All of the reported results are in percentages (%).

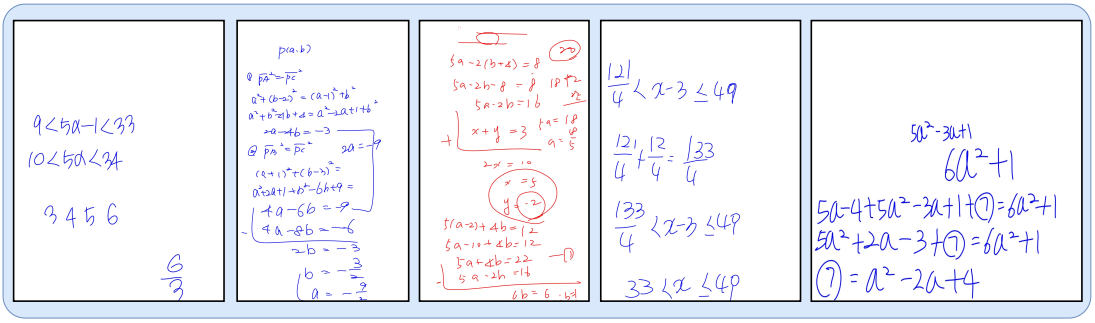

Figure 11: Sample data from the test set of the AIHub dataset.

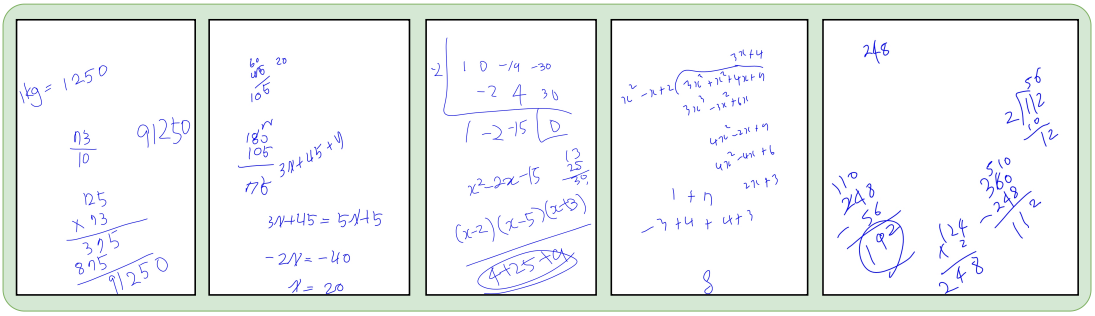

Figure 12: Examples containing heavily rotated mathematical expressions from Challenging Subset of AIHub.

all three education levels. It achieves 72.10% / 48.60% (ED) and 60.04% / 59.85% (EL) at the Primary School level, climbing to 73.40% / 49.35% (ED) and 63.83% / 63.45% (EL) at the High School level. This trend demonstrates that VEHME not only generalizes well across increasing difficulty but also narrows the gap with closed-source systems in EL, where precise localization becomes more challenging.

Overall, our findings suggest that performance is largely consistent across grade levels, with no significant degradation or sensitivity to problem complexity. This indicates that the models' ability to detect and localize errors is not strongly dependent on the education level of the task.

A.6 Challenging Subset of AIHub

In our analysis, most of the standard dataset has mild rotation ($< 15°$) (as shown in Figure 11), which might explain the relatively small global improvement from EVPM. To better understand EVPM's contribution, we constructed a "*Challenging Subset of AIHub*" by identifying the 100 handwritten data points with the most heavily rotated mathematical expressions (top-100; Mean: 21.81°, Std: 3.40°, Min: 17.56°, Max: 33.20°) as shown in the Figure 12.

A.7 Qualitative Result

We further compare our qualitative results against other baselines, as shown in Figure 13 and Figure 14.

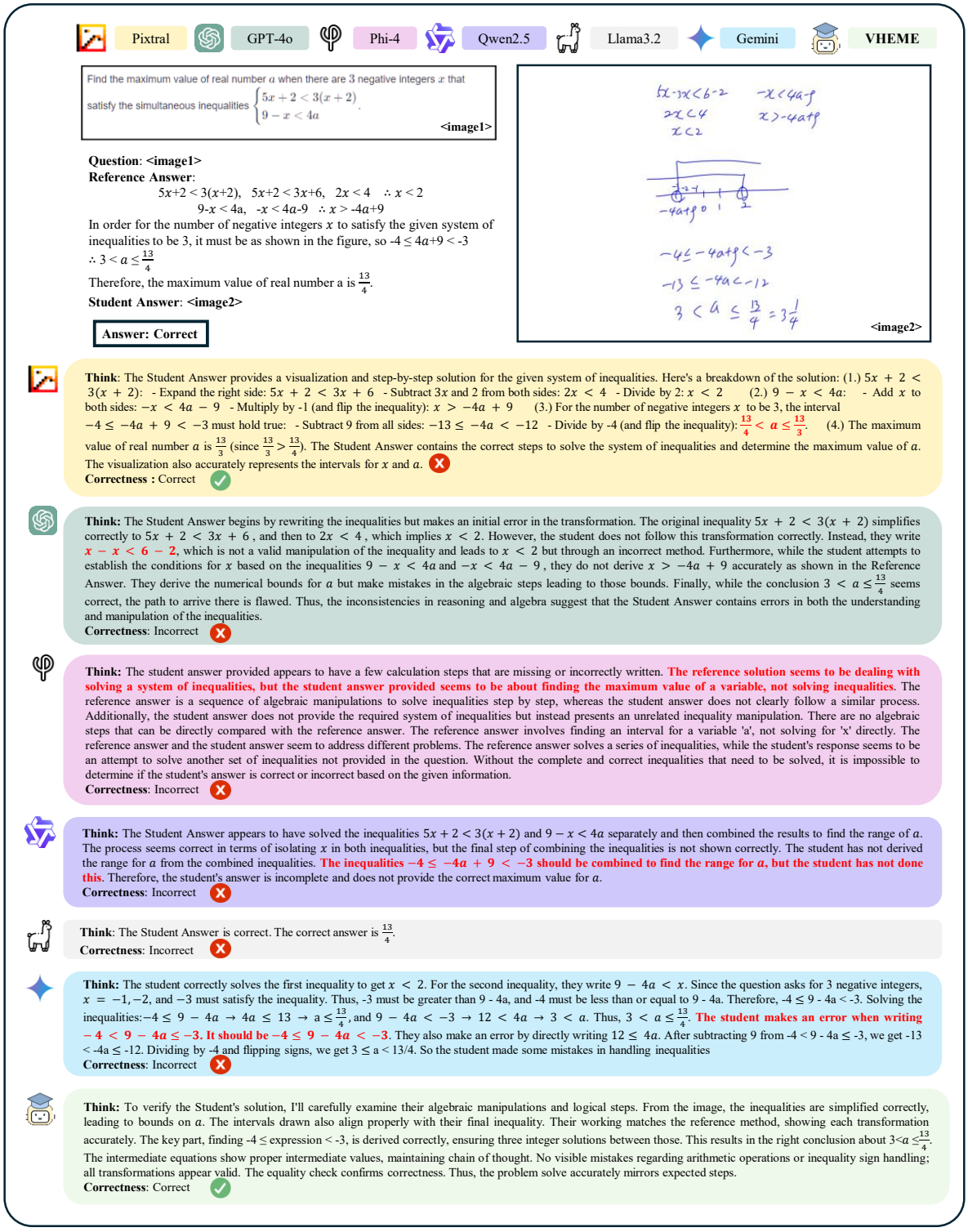

Figure 13: Qualitative example for Error Detection (ED). Among the 7 models, VEHME provides the most correct error detection. The model's error is colored in red. The incorrect response from the models can be categorized into 3 main types: "*recognition error*", "*problem misunderstanding*", and "hallucination". Pixtral, GPT, Qwen2.5, and Gemini all suffer from incorrect OCR of the student's handwritten mathematical expressions. While Pixtral still produces the correct assessment, its thought was wrong due to incorrect OCR. Phi-4 misinterprets the problem as solving the inequality, while the problem is asking to find the maximum value of a. Llama3.2 suffers from hallucination where its thought is correct, but the final assessment is *incorrect*.

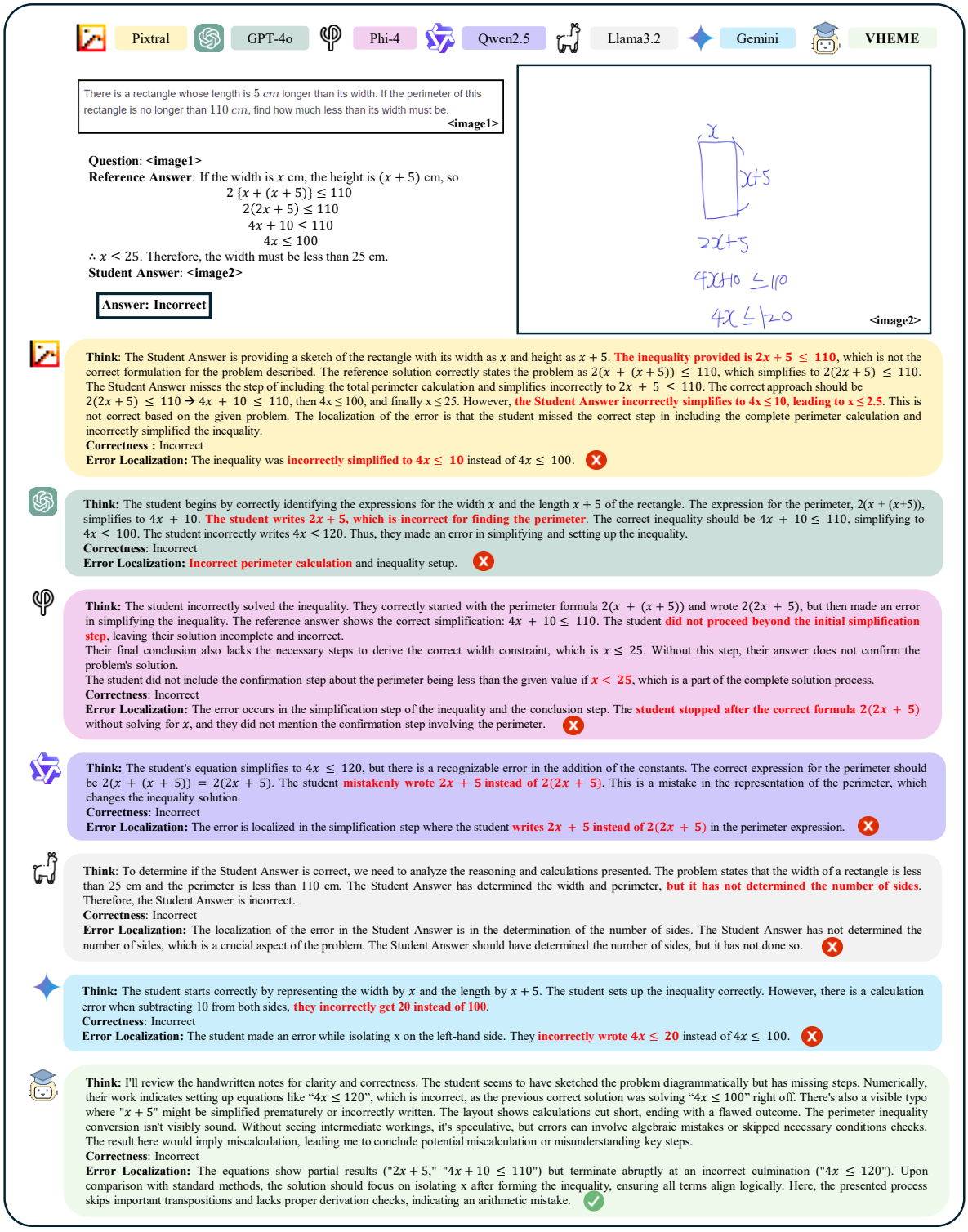

Figure 14: Qualitative example for Error Localization (EL). Among the 7 models, VEHME provides the most correct localization. The model's error is colored in red. The incorrect response from the models can be categorized into 2 main types: "*recognition error*" and "*student's answer misunderstanding*". Pixtral, Phi-4, and Gemini all suffer from incorrect OCR of the student's handwritten mathematical expressions. Pixtral and Gemini misread the inequality $4x \leq 120$ while Phi-4 cannot read beyond the inequality $2x + 5$. GPT, Qwen2.5, and Llama3.2 misinterpret the student's answer. GPT and Qwen2.5 think that the student writes $2x + 5$ as the perimeter, while Llama3.2 thinks that the student must determine the number of sides.