EXAMINKBENCH: BENCHMARKING END-TO-END LINE-LEVEL STUDENT ANSWER RECONSTRUCTION FROM COMPLETE EXAM PAGES

Anonymous authors
Paper under double-blind review

ABSTRACT

Reliable automated assessment requires reconstructing students' complete responses from full examination pages rather than transcribing isolated handwriting. Existing benchmarks typically evaluate handwriting recognition, document parsing, and answer grounding separately, making coupled failure modes difficult to measure. Therefore, we curate a gold-standard corpus of 4,017 fully annotated samples from approximately 31,769 real mathematics examination papers, with 141 public samples forming **ExamInkBench** and 3,876 reserved for internal development and self-evolution. Each response is represented as an ordered sequence of answer lines grouped by question item. We design a two-level evaluation protocol that separately measures line-level extraction and ordering, as well as intra-line transcription fidelity. Evaluations of mainstream multimodal models reveal frequent answer-line omissions and incorrect question associations, while accurate local recognition does not guarantee correct page-level reconstruction. Motivated by these structured failures, we propose **ExamInk-Evo**, a failure-driven skill self-evolution method that mines recurrent errors, induces targeted reconstruction and verification skills, and retains updates only when held-out validation performance improves. Experiments with tool-using agents show that the method consistently improves both extraction and transcription, with larger gains under stricter line-matching thresholds. The evolved skills also transfer positively across agents, indicating that they capture reusable reconstruction procedures rather than agent-specific behaviors.

1 INTRODUCTION

Recent advances in multimodal large language models and document-understanding systems have substantially improved visual question answering, handwriting recognition, and page-level document parsing (Li et al., 2023; Huang et al., 2022; Kim et al., 2022; Mathew et al., 2021; Lu et al., 2024). These developments make automated educational assessment increasingly practical. However, existing automated grading studies often assume that student responses have already been correctly segmented and transcribed (Ke & Ng, 2019; Gold & Zesch, 2020; e Silva et al., 2024). In real examination workflows, a reliable system must instead recover every student-written answer line from a complete examination page, associate each line with the correct question or subquestion, preserve its original order, and faithfully transcribe both natural language and mathematical expressions. We refer to this end-to-end capability as **student answer reconstruction**.

Complete examination pages introduce structural difficulties that are absent from isolated handwriting crops. Erasures, overwriting, and irregular layouts can obscure line boundaries and disrupt reading order (Huang et al., 2023; Wang et al., 2021). These errors are tightly coupled. Omitting one line may remove a critical reasoning step, assigning a line to the wrong question may change the answer structure, and reordering lines may invalidate a derivation even when the individual lines are transcribed correctly. Handwritten-answer scoring studies further show that recognition errors can propagate to semantic grading decisions (Gold & Zesch, 2020). As illustrated in Fig. 1, a local recognition error can propagate through question association and answer-structure reconstruction, ultimately leading to an incorrect grading decision.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

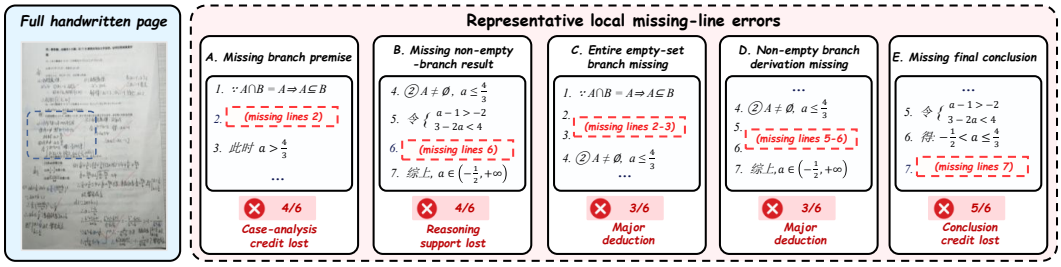

Figure 1: Illustration of error propagation in student answer reconstruction. A local handwriting recognition error can lead to an incorrect question association or a missing reasoning step, which then affects the reconstructed answer structure and the final semantic grading decision.

To understand why strong component-level scores do not guarantee reliable student answer reconstruction, we examine existing benchmark datasets. These benchmarks largely evaluate complementary capabilities in isolation. BNU-Exam-Benchmark and OmniHandwritingOCR focus on handwriting transcription under realistic writing conditions, OmniDocBench evaluates document parsing and reading structure, and OCRBench, OCRBench v2, and educational grounding benchmarks assess OCR, localization, or answer grounding (Li et al., 2026; Guo et al., 2026; Ouyang et al., 2025; Liu et al., 2024; Fu et al., 2026; Zhao et al., 2026). DocVQA evaluates structure-sensitive visual question answering over document images (Mathew et al., 2021), whereas MathWriting focuses on isolated handwritten mathematical expressions (Gervais et al., 2025). These settings do not jointly require exhaustive recovery of all student answers from a complete examination page while preserving question or subquestion identity, original answer-line order, and transcription fidelity. A system may therefore achieve strong recognition or parsing scores while still omitting answer lines, misassigning them to questions, disrupting derivations, or distorting mathematical expressions.

To address this **gap**, we introduce **ExamInkBench**, a gold-standard benchmark for end-to-end student answer reconstruction from complete examination pages. We fully annotate 4,017 valid samples. Among them, 141 publicly released samples constitute ExamInkBench, the public evaluation set, while the remaining 3,876 samples are reserved for internal development and self-evolution. The subsets are disjoint. Student responses are annotated by question and subquestion, with answer lines as the atomic units and their original order preserved. We develop a structure-aware evaluation protocol that measures question-aware answer-line coverage using Precision, Recall, and F1, line-order preservation using Line Order Accuracy, and textual and mathematical transcription fidelity using CER, ROUGE-L, and LaTeX Token Error Rate (Lin, 2004; Deng et al., 2017).

We evaluate recent multimodal models on **ExamInkBench**. The results show that models frequently omit answer lines, associate them with incorrect questions, disrupt their original order, and produce inaccurate textual or mathematical transcriptions. In many cases, individual lines are recognized correctly, but the reconstructed answer remains incorrect because its coverage or structure is incomplete. Line extraction and transcription quality are also not always aligned, and OCR-oriented systems do not by themselves resolve the page-level reconstruction problem. These results confirm that **ExamInkBench** exposes failure modes that are not captured by isolated handwriting recognition or generic document parsing, while providing structured diagnostic signals for system improvement.

The structured nature of these failures motivates an agent-based optimization route. An agentic system can retain intermediate predictions and execution traces, inspect reconstruction errors, and perform targeted verification before producing its final output. Based on this observation, we propose **ExamInk-Evo**, a failure-driven skill self-evolution method. Starting from an agent program without task-specific skills, the method mines recurrent failure patterns from the training split and converts them into targeted reconstruction and verification skills. A candidate update is retained only when it improves performance on an independent held-out development split. Our contributions are summarized as follows:

- We construct an ExamInk corpus of 4,017 fully annotated samples, of which 141 public samples form **ExamInkBench** and 3,876 samples are retained for internal development and self-evolution.

- We develop a **structure-aware evaluation protocol** covering question-aware answer-line extraction, line ordering, and intra-line textual and mathematical transcription. We apply this protocol to mainstream multimodal models and tool-using agents, systematically characterizing their coverage, question-association, ordering, and transcription failures.
- We propose **ExamInk-Evo**, a failure-driven skill self-evolution framework that converts fine-grained reconstruction failures into reusable skills for tool-using agents. The framework validates candidate skill updates on held-out data and improves student answer reconstruction across both self-evolution and cross-agent transfer settings.

2 Related Work

Visual Text Understanding in VLMs. Recent VLMs have substantially improved visual text recognition, document understanding, and multimodal reasoning. Their capabilities now extend beyond image-level question answering to OCR, mathematical expression recognition, layout analysis, structured information extraction, and long-context document processing. Representative model families include Qwen, GLM, DeepSeek, Kimi, GPT, Gemini, and Claude, covering omni-modal, OCR-oriented, agentic, and general-purpose VLMs (Team, 2026; Duan et al., 2026; Xu et al., 2026). However, these capabilities are commonly evaluated through isolated text recognition, document question answering, layout parsing, or structured extraction tasks (Mathew et al., 2021; Liu et al., 2024; Ouyang et al., 2025). A model may therefore transcribe visible text correctly while still failing to recover all answer lines, associate them with the correct questions, preserve their original order, or reconstruct the complete reasoning structure. ExamInkBench evaluates these capabilities jointly on complete handwritten examination pages.

Handwritten and Educational Benchmarks. Previous benchmarks have largely targeted specific handwritten or document-understanding scenarios. IAM, RIMES, and SCUT-HCCDoc evaluate offline handwriting recognition under different writing conditions (Marti & Bunke, 2002; Grosicki et al., 2009; Zhang et al., 2020), while MathWriting focuses on handwritten mathematical expression recognition (Gervais et al., 2025). Examination-oriented benchmarks such as BNU-Exam-Benchmark, EnsExam, and OmniHandwritingOCR introduce realistic exam artifacts, handwriting noise, and formula transcription challenges (Li et al., 2026; Huang et al., 2023; Guo et al., 2026). Other datasets study page-level document understanding, including form parsing in FUNSD, layout analysis in DocLayNet, document question answering in DocVQA, and multi-level PDF parsing in OmniDocBench (Jaume et al., 2019; Pfitzmann et al., 2022; Mathew et al., 2021; Ouyang et al., 2025). OCRBench and OCRBench v2 further combine multiple OCR-oriented tasks, while HG-Bench investigates handwritten answer-region grounding (Liu et al., 2024; Fu et al., 2026; Zhao et al., 2026). Despite this progress, existing benchmarks generally evaluate recognition, parsing, localization, or grounding as separate tasks. They do not jointly require exhaustive recovery of all student answer lines from a complete examination page, correct question or subquestion association, original line ordering, and faithful transcription. ExamInkBench is designed to fill this gap.

3 Benchmark Construction

This section defines the student answer reconstruction task, describes data curation and annotation, summarizes benchmark statistics, and presents the evaluation protocol.

3.1 Task Description

Student answer reconstruction. Given a complete examination image X, the system must identify question items and reconstruct the student's answers as ordered lines, jointly performing answer-line extraction, question association, line ordering, and textual or mathematical transcription. The gold answer structure is defined as:

$$Y = \{(q, G_q) : q \in Q\}, \qquad G_q = (g_{q,1}, \ldots, g_{q,n_q}), \tag{1}$$

where Q is the set of annotated question or subquestion identifiers and G_q is the ordered sequence of answer lines associated with question q. Unanswered questions remain in Q with $G_q = ()$.

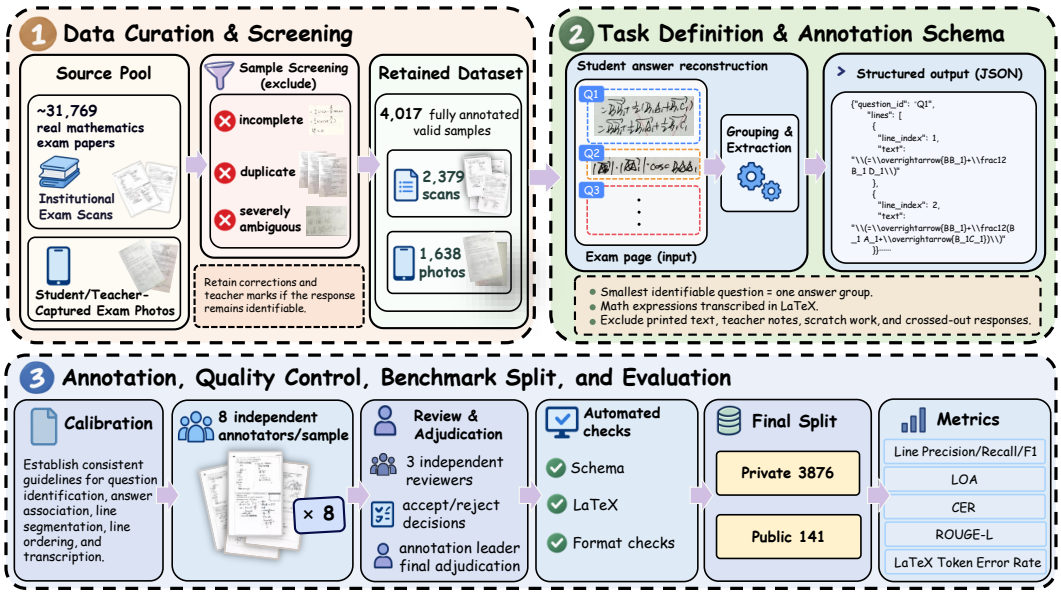

Figure 2: Annotation workflow for EXAMINKBENCH. The process includes guideline calibration, independent annotation, expert review, adjudication, and automated quality validation.

The system prediction is:

$$\widehat{Y} = \{(q, P_q) : q \in \widehat{Q}\}, \qquad P_q = (p_{q,1}, \ldots, p_{q,m_q}), \tag{2}$$

where $\widehat{Q}$ is the predicted question set and P_q is the predicted answer-line sequence for question q.

3.2 DATA CURATION AND ANNOTATION

From 31,769 real mathematics examination papers, we exclude damaged and severely ambiguous samples and fully annotate 4,017 valid samples, including 2,379 institutional scans and 1,638 student- or teacher-captured photographs. The 141 public samples constitute **ExamInkBench**, while the remaining samples are used for internal model development and self-evolution. For each retained sample, an answer group corresponds to the smallest identifiable question or subquestion, and its answer lines follow the visible writing order. Each group contains a `question_id` and an ordered sequence of `line_index–text` pairs, with mathematical expressions represented in LaTeX. Unanswered questions are represented by empty answer sequences.

The annotation workflow includes calibration, independent annotation, expert review, adjudication, and automated validation, as illustrated in Fig. 2. Calibration standardizes question identification, answer association, line segmentation, ordering, and transcription. Each sample is independently annotated by eight annotators and reviewed against the original image by three additional reviewers. An annotation leader resolves disagreements before automated structural and format checks. Annotation consistency evaluation is provided in Appendix A.

3.3 BENCHMARK STATISTICS

The annotated collection contains 29,151 answer groups and 124,706 student answer lines (Fig. 3). Each page contains 1–24 groups, each group contains 0–28 lines, and page-level counts concentrate at 25–29 lines while extending to 80–84 lines. Math-only lines account for 58.7% of all lines and mixed text-and-mathematics lines for 35.0%, requiring systems to handle both short responses and long derivations while preserving question association and order.

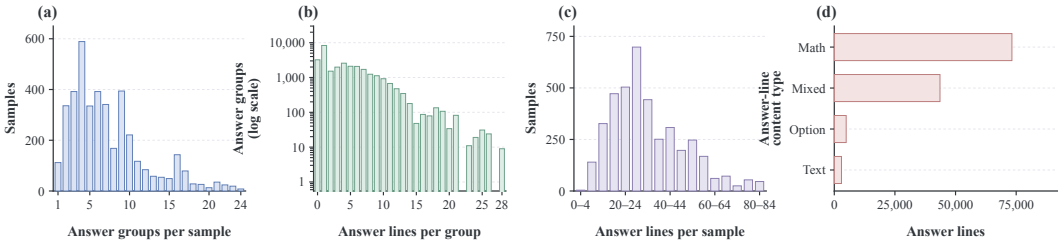

Figure 3: Statistics of the annotated EXAMINKBENCH collection, showing groups per page, lines per group and page, and answer-line content types.

3.4 EVALUATION CRITERIA

Our evaluation follows two hierarchical levels. The first measures line-level extraction, including question-aware matching, coverage, and ordering. The second evaluates intra-line transcription fidelity for textual and mathematical content.

Line-level extraction accuracy. For a gold line $g_{q,i}$ and a predicted line $p_{q,j}$ assigned to the same question q, we compute their normalized similarity as:

$$s_{ij}^{(q)} = 1 - \frac{\mathrm{Lev}(\phi(g_{q,i}), \phi(p_{q,j}))}{\max(|\phi(g_{q,i})|, |\phi(p_{q,j})|)}, \tag{3}$$

where Lev denotes the Levenshtein distance and ϕ is a shared normalization function. Hungarian matching is performed independently for each question.

Let K be the total number of valid matches, and let N_{pred} and N_{gt} be the numbers of predicted and gold answer lines. Answer-line precision, recall, and F1 are defined as:

$$P_{\mathrm{line}} = \frac{K}{N_{\mathrm{pred}}}, \qquad R_{\mathrm{line}} = \frac{K}{N_{\mathrm{gt}}}, \qquad F1_{\mathrm{line}} = \frac{2K}{N_{\mathrm{pred}} + N_{\mathrm{gt}}}. \tag{4}$$

For each question q, let M_q be its valid matches, $K_q = |M_q|$, and $T_q = \binom{K_q}{2}$. For two matches $(i,j),(i',j') \in M_q$ with $i < i'$, the predicted order is correct when $j < j'$. If C_q is the number of correctly ordered pairs, Line Order Accuracy is:

$$\mathrm{LOA} = \frac{\sum_q C_q}{\sum_q T_q}. \tag{5}$$

Intra-line transcription accuracy. For each valid match (g,p), edit operations are computed on normalized text or LaTeX tokens. Unmatched gold lines count as deletions and receive a ROUGE-L F1 score of zero, while unmatched predictions count as insertions. Textual transcription is measured by the Character Error Rate:

$$\mathrm{CER} = \frac{S_C + D_C + I_C}{N_{\mathrm{char}}^{\mathrm{GT}}}. \tag{6}$$

where S_C, D_C, and I_C are character substitutions, deletions, and insertions aggregated over all gold lines, and $N_{\mathrm{char}}^{\mathrm{GT}}$ is the total number of gold characters. ROUGE-L F1 (Lin, 2004) is computed over normalized character sequences and averaged over all gold lines. Mathematical transcription is evaluated using the LaTeX Token Error Rate:

$$\mathrm{TER}_{\mathrm{LaTeX}} = \frac{S_T + D_T + I_T}{N_{\mathrm{token}}^{\mathrm{GT}}}. \tag{7}$$

where S_T, D_T, and I_T are token substitutions, deletions, and insertions over all gold lines, and $N_{\mathrm{token}}^{\mathrm{GT}}$ is the number of gold LaTeX tokens. Further details are provided in Appendix B.

4 RESULTS AND ANALYSIS

We evaluate 11 multimodal systems from the Claude, DeepSeek, Gemini, GLM, GPT, Kimi, and Qwen families on the 141-sample ExamInkBench set and analyze limitations in full-page student-answer reconstruction. They cover general-purpose, omni-modal, and OCR-oriented models. See Appendix C for details.

Table 1: Line-level extraction and intra-line transcription results under different line-matching thresholds τ_{line}. Bold and underlined values indicate the best and second-best results, respectively. ↑ indicates that higher is better, and ↓ indicates that lower is better.

(a) Line-level Extraction Metrics

System	$\tau_{\text{line}} = 0.4$				$\tau_{\text{line}} = 0.5$				$\tau_{\text{line}} = 0.6$			
	LineF1↑	P↑	R↑	LOA↑	LineF1↑	P↑	R↑	LOA↑	LineF1↑	P↑	R↑	LOA↑
Claude-Fable-5-1	0.6112	0.5827	0.6427	0.4610	0.5123	0.4884	0.5386	0.3418	0.3811	0.3634	0.4008	0.2082
DeepSeek-Flash	0.4766	0.4311	0.5328	0.3373	0.3710	0.3356	0.4147	0.2262	0.2527	0.2286	0.2825	0.1201
Gemini-3.8-Flash	0.6848	0.6505	0.7229	0.5507	0.5482	0.5208	0.5787	0.3719	0.3835	0.3643	0.4048	0.1997
GLM-4.6V	0.3944	0.4071	0.3825	0.2009	0.2908	0.3002	0.2820	0.1201	0.1919	0.1980	0.1861	0.0645
GLM-5.3-Flash	0.6401	0.6127	0.6700	0.5046	0.5296	0.5070	0.5544	0.3732	0.3928	0.3760	0.4111	0.2267
GLM-OCR	0.1725	0.1567	0.1919	0.1041	0.1085	0.0986	0.1207	0.0483	0.0545	0.0495	0.0606	0.0153
GPT-5.6-sol	0.6864	0.6660	0.7080	0.5401	**0.6038**	**0.5859**	**0.6229**	**0.4360**	**0.4807**	**0.4664**	**0.4958**	**0.3001**
Kimi-K2.6	0.6474	0.5845	0.7254	0.5663	0.5358	0.4838	0.6004	0.4065	0.3967	0.3581	0.4445	0.2441
Kimi-K3	**0.7000**	**0.6683**	**0.7349**	**0.5803**	0.5880	0.5614	0.6173	0.4237	0.4309	0.4114	0.4524	0.2440
Qwen3.5-Omni-Plus	0.5903	0.5777	0.6035	0.4373	0.4707	0.4587	0.4832	0.3036	0.3206	0.3138	0.3278	0.1565
Qwen3.8-Omni-Flash	0.6694	0.6380	0.7040	0.5178	0.5591	0.5329	0.5880	0.3815	0.4160	0.3965	0.4375	0.2312

(b) Intra-Line Transcription Metrics

System	$\tau_{\text{line}} = 0.4$			$\tau_{\text{line}} = 0.5$			$\tau_{\text{line}} = 0.6$		
	ROUGE-L↑	CER↓	TER↓	ROUGE-L↑	CER↓	TER↓	ROUGE-L↑	CER↓	TER↓
Claude-Fable-5-1	0.4825	0.9581	0.8673	0.4211	1.1089	1.0303	0.3293	1.3015	1.2381
DeepSeek-Flash	0.3923	1.3558	1.2527	0.3216	1.5235	1.4194	0.2325	1.7318	1.6226
Gemini-3.8-Flash	0.5382	0.9397	0.7480	0.4509	1.1517	0.9614	0.3326	1.4387	1.2668
GLM-4.6V	0.2826	1.2210	1.0168	0.2219	1.3443	1.0814	0.1557	1.4879	1.1680
GLM-5.3-Flash	0.5044	0.8653	0.8054	0.4359	1.0194	0.9479	0.3394	1.2230	1.1457
GLM-OCR	0.1323	2.5073	1.7096	0.0895	2.6360	1.7757	0.0483	2.7544	1.8484
GPT-5.6-sol	0.5487	**0.7103**	**0.6148**	**0.4989**	**0.8171**	**0.7242**	**0.4145**	**0.9881**	**0.9027**
Kimi-K2.6	0.5473	0.9727	0.8405	0.4727	1.1466	1.0228	0.3674	1.3875	1.2868
Kimi-K3	**0.5549**	0.8442	0.7012	0.4850	1.0072	0.8690	0.3737	1.2415	1.1342
Qwen3.5-Omni-Plus	0.4504	1.0035	0.8545	0.3751	1.1835	0.9857	0.2713	1.4068	1.1830
Qwen3.8-Omni-Flash	0.5333	0.8839	0.7735	0.4646	1.0385	0.9340	0.3633	1.2629	1.1750

4.1 Main Experimental Results

Table 1 reports line-level extraction and intra-line transcription metrics under three matching thresholds, $\tau_{\text{line}} \in \{0.4, 0.5, 0.6\}$. At the line level, Kimi-K3 performs best at $\tau_{\text{line}} = 0.4$, achieving a Line F1 of 0.7000 and the highest precision, recall, and LOA. GPT-5.6-sol follows with a Line F1 of 0.6864. At $\tau_{\text{line}} = 0.5$ and 0.6, GPT-5.6-sol leads all line-level metrics, with Line F1 scores of 0.6038 and 0.4807, respectively.

Intra-line and extraction rankings diverge. At $\tau_{\text{line}} = 0.4$, Kimi-K3 obtains the highest ROUGE-L score of 0.5549, whereas GPT-5.6-sol achieves the lowest CER and $\text{TER}_{\text{LaTeX}}$, at 0.7103 and 0.6148. At the stricter thresholds, GPT-5.6-sol leads all three transcription metrics. Thus, stronger answer-line extraction does not imply higher transcription accuracy. Across systems, increasing τ_{line} consistently lowers Line F1, LOA, and ROUGE-L while increasing CER and $\text{TER}_{\text{LaTeX}}$.

4.2 Key Findings

Finding 1: Full-page answer reconstruction remains challenging. Even the best-performing models cannot completely and reliably recover all student answers. When $\tau_{\text{line}} = 0.6$, the best Line F1 is only 0.4807, and the best LOA is only 0.3001. The clear performance decline from $\tau_{\text{line}} = 0.4$ to 0.6 indicates that many predicted lines are only borderline matches and are judged as omissions when higher matching confidence is required.

Finding 2: Structural reconstruction is weaker than answer-line coverage. Line-order performance is consistently lower than answer-line coverage. For example, at $\tau_{\text{line}} = 0.4$, Kimi-K3 achieves a Line F1 of 0.7000 but an LOA of only 0.5803, while GPT-5.6-sol obtains values of 0.6864 and 0.5401. At $\tau_{\text{line}} = 0.6$, GPT-5.6-sol achieves a Line F1 of 0.4807, but its LOA decreases to 0.3001. Because answer-line matching is question-aware, these gaps show that recognizing individual answer lines does not guarantee correct answer structure or original ordering.

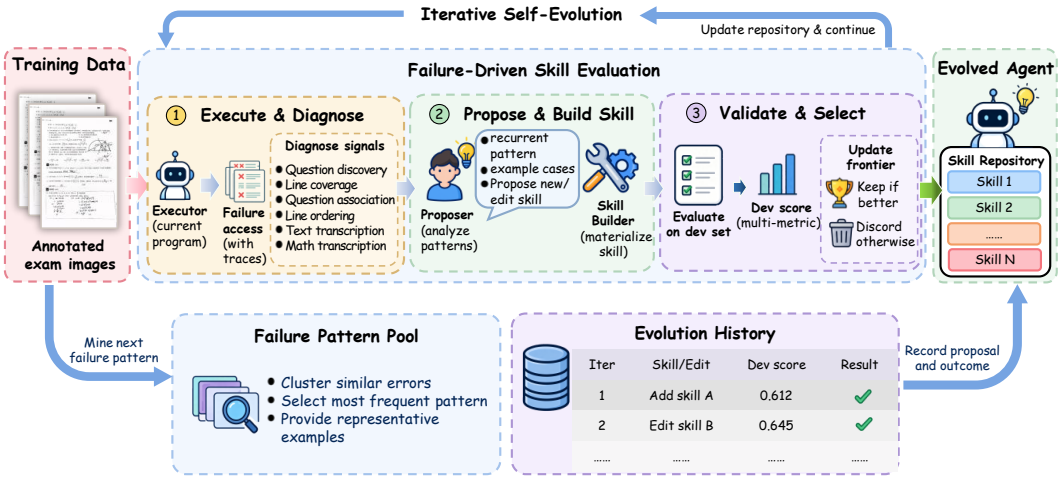

Figure 4: Overview of EXAMINK-EVO. Recurrent reconstruction failures are mined from execution results, converted into targeted skills, and evaluated on held-out data before retention.

Finding 3: Answer-line extraction and transcription are decoupled. At $\tau_{\text{line}} = 0.4$, Kimi-K3 leads in Line F1 and ROUGE-L, whereas GPT-5.6-sol achieves lower CER and LaTeX Token Error Rate. Despite its slightly lower Line F1 (0.6864 vs. 0.7000), GPT-5.6-sol obtains a much lower CER (0.7103 vs. 0.8442). Thus, extraction and transcription should be evaluated separately.

Finding 4: OCR capability alone is insufficient for answer reconstruction. GLM-OCR consistently trails general-purpose multimodal models on line-level and intra-line metrics, especially recall and line ordering. Thus, local text recognition alone cannot recover all answers, associate them with the correct questions, or preserve their order on complete examination pages.

5 EXAMINK-EVO: FAILURE-DRIVEN SKILL SELF-EVOLUTION FOR STUDENT ANSWER RECONSTRUCTION

We propose EXAMINK-EVO, a failure-driven skill self-evolution method for student answer reconstruction. The framework mines recurrent failures, induces targeted reconstruction and verification skills, and retains only updates that improve held-out performance, as summarized in Figure 4.

5.1 FRAMEWORK OVERVIEW

An agent program p combines a base agent with a task-specific skill repository. Evolution starts without task-specific skills and uses three roles: an **Executor**, a **Proposer**, and a **Skill Builder**.

Executor. The Executor $\mathcal{E}$ runs the current program on complete examination pages and retains predictions and execution traces for failure analysis.

Proposer. The Proposer $\mathcal{P}$ analyzes recurrent failures, execution traces, gold annotations, the current repository, and evolution history to propose a new skill or modify an existing one.

Skill Builder. The Skill Builder $\mathcal{B}$ converts each proposal into a structured skill with trigger conditions and procedural instructions, optionally incorporating helper scripts or reference material. The resulting skill is combined with the parent program to form a candidate program.

5.2 DIAGNOSTIC FAILURE MINING

Using the question-aware matching procedure in Sec. 3.4, we compare each prediction $\widehat{Y}$ with its gold structure Y and summarize the resulting diagnostics as:

$$\mathcal{D}(x; p) = (D_{\text{ques}}, D_{\text{cov}}, D_{\text{assoc}}, D_{\text{order}}, D_{\text{text}}, D_{\text{math}}), \tag{8}$$

where the six components represent question discovery, answer-line coverage, question association, line ordering, textual transcription, and mathematical transcription errors, respectively.

Diagnostics combine matching correspondences and edit operations. Unmatched predictions are compared with gold lines from other questions, with high cross-question similarity indicating misassociation. We cluster recurrent patterns over the training split and select the most frequent pattern at each iteration, breaking ties by error magnitude. Its representative examples and execution traces form the Proposer's failure set $\mathcal{F}_t$.

5.3 Evolution Loop

We maintain a bounded frontier $\mathcal{G}$ of candidate programs and a cumulative evolution history $\mathcal{H}$. At iteration t, a parent program p_t is selected from $\mathcal{G}$. Given the mined failure set $\mathcal{F}_t$, the Proposer generates one skill mutation $\pi_t = \mathcal{P}\left(\mathcal{F}_t, \mathcal{H}, \mathrm{Skills}(p_t)\right)$.

The Skill Builder then produces a candidate program:

$$\widetilde{p}_t = \mathcal{B}(p_t, \pi_t). \tag{9}$$

The candidate is evaluated on the held-out development split. We first clip the two error metrics:

$$\overline{\mathrm{CER}} = \min(\mathrm{CER}, 1), \qquad \overline{\mathrm{TER}}_{\mathrm{LaTeX}} = \min(\mathrm{TER}_{\mathrm{LaTeX}}, 1), \tag{10}$$

and define the development objective as the equally weighted combination:

$$\mathcal{S}_{\mathrm{dev}}(p) = \frac{1}{5}\left[F1_{\mathrm{line}} + \mathrm{LOA} + \text{ROUGE-L} + \left(1 - \overline{\mathrm{CER}}\right) + \left(1 - \overline{\mathrm{TER}}_{\mathrm{LaTeX}}\right)\right]. \tag{11}$$

Only the aggregate development score is exposed during candidate selection. If the frontier has not reached its capacity k, the candidate is added directly. Otherwise, it is retained only when:

$$\mathcal{S}_{\mathrm{dev}}(\widetilde{p}_t) > \min_{p \in \mathcal{G}} \mathcal{S}_{\mathrm{dev}}(p), \tag{12}$$

in which case the lowest-scoring frontier member is removed. Each proposal and its validation outcome are appended to $\mathcal{H}$, regardless of whether the candidate is retained. The process runs for at most T iterations, and the final program is selected as:

$$p^* = \arg\max_{p \in \mathcal{G}} \mathcal{S}_{\mathrm{dev}}(p). \tag{13}$$

5.4 Experiments

We evaluate **ExamInk-Evo** through self-evolution and cross-agent transfer. For self-evolution, the 3,876 private samples are split approximately 8:2 into self-evolution and held-out validation subsets. Final self-evolution and transfer results are evaluated on the fixed 141-sample public ExamInkBench. Implementation details for the self-evolution experiments are provided in Appendix D. For transfer, a frozen skill repository evolved by one agent is applied to another without further updates. Results under all three matching thresholds are reported in Tables 2 and 3.

Self-evolution results. Table 2 shows that ExamInk-Evo consistently improves both line-level extraction and intra-line transcription. For WorkBuddy, the LineF1 improvement increases from 5.6% to 9.1% as τ_{line} increases from 0.4 to 0.6, while the LOA improvement grows from 10.3% to 17.0%. For Codex, the gains of LineF1 and LOA rise progressively across the three thresholds. The stronger gains under stricter matching indicate that the evolved skills improve the recovery of difficult answer lines and their ordering.

Transcription also improves: WorkBuddy's ROUGE-L gains are 9.4%–12.3%, versus 8.6%–12.4% for Codex. For Codex at $\tau_{\mathrm{line}} = 0.4$, CER and LaTeX token error rate decrease by 21.9% and 22.9%, respectively. Thus, failure-driven updates improve extraction and transcription jointly.

Cross-agent skill transfer. Table 3 shows positive transfer across agents. WorkBuddy's repository improves Codex's LineF1 and LOA by 2.7%–4.6% and 6.5%–11.0%, respectively, while Codex's repository improves WorkBuddy's by 5.4%–8.5% and 9.9%–16.3%. Transcription also improves in both directions, with higher ROUGE-L and lower CER and LaTeX token error rates. These results suggest that the skills capture reusable procedures for identifying, associating, ordering, and verifying answer lines. Asymmetric gains indicate target-dependent transfer, while bidirectional improvements show generalization beyond the source agent.

Table 2: Line-level extraction and intra-line transcription results under different τ_{line}. Δ reports relative improvement over the corresponding baseline. ↑: higher is better; ↓: lower is better.

(a) Line-level Extraction Metrics

System	$\tau_{\text{line}} = 0.4$				$\tau_{\text{line}} = 0.5$				$\tau_{\text{line}} = 0.6$			
	LineF1↑	P↑	R↑	LOA↑	LineF1↑	P↑	R↑	LOA↑	LineF1↑	P↑	R↑	LOA↑
WorkBuddy (DeepSeek-V4.1-Flash)	0.7671	0.7738	0.7605	0.5940	0.6883	0.6942	0.6824	0.4937	0.5783	0.5833	0.5733	0.3710
+ ExamInk-Evo	0.8104	0.7960	0.8254	0.6550	0.7344	0.7213	0.7479	0.5569	0.6311	0.6198	0.6427	0.4340
Δ (%)	**+5.6%**	**+2.9%**	**+8.5%**	**+10.3%**	**+6.7%**	**+3.9%**	**+9.6%**	**+12.8%**	**+9.1%**	**+6.3%**	**+12.1%**	**+17.0%**
Codex (GPT-5.6-sol)	0.8535	0.8432	0.8639	0.7273	0.7989	0.7894	0.8087	0.6393	0.7095	0.7010	0.7182	0.5187
+ ExamInk-Evo	0.8966	0.8841	0.9094	0.8146	0.8542	0.8423	0.8664	0.7420	0.7791	0.7683	0.7903	0.6363
Δ (%)	**+5.1%**	**+4.9%**	**+5.3%**	**+12.0%**	**+6.9%**	**+6.7%**	**+7.1%**	**+16.1%**	**+9.8%**	**+9.6%**	**+10.0%**	**+22.7%**

(b) Intra-line Transcription Metrics

System	$\tau_{\text{line}} = 0.4$			$\tau_{\text{line}} = 0.5$			$\tau_{\text{line}} = 0.6$		
	ROUGE-L↑	CER↓	TER↓	ROUGE-L↑	CER↓	TER↓	ROUGE-L↑	CER↓	TER↓
WorkBuddy (DeepSeek-V4.1-Flash)	0.6428	0.6780	0.6128	0.5949	0.8069	0.7380	0.5185	0.9955	0.9268
+ ExamInk-Evo	0.7035	0.6155	0.5866	0.6558	0.7440	0.7053	0.5823	0.9196	0.8699
Δ (%)	**+9.4%**	**+9.2%**	**+4.3%**	**+10.2%**	**+7.8%**	**+4.4%**	**+12.3%**	**+7.6%**	**+6.1%**
Codex (GPT-5.6-sol)	0.7425	0.3658	0.2950	0.7102	0.4281	0.3612	0.6501	0.5397	0.4816
+ ExamInk-Evo	0.8062	0.2857	0.2275	0.7809	0.3345	0.2788	0.7304	0.4286	0.3793
Δ (%)	**+8.6%**	**+21.9%**	**+22.9%**	**+10.0%**	**+21.9%**	**+22.8%**	**+12.4%**	**+20.6%**	**+21.2%**

Table 3: Cross-agent skill transfer results under different τ_{line}. Codex (GPT-5.6-sol) uses WorkBuddy's skill, and WorkBuddy (DeepSeek-V4.1-Flash) uses Codex's skill. Δ reports relative improvement over the corresponding baseline. ↑: higher is better; ↓: lower is better.

(a) Line-level Extraction Metrics

System	$\tau_{\text{line}} = 0.4$				$\tau_{\text{line}} = 0.5$				$\tau_{\text{line}} = 0.6$			
	LineF1↑	P↑	R↑	LOA↑	LineF1↑	P↑	R↑	LOA↑	LineF1↑	P↑	R↑	LOA↑
Codex (GPT-5.6-sol)	0.8535	0.8432	0.8639	0.7273	0.7989	0.7894	0.8087	0.6393	0.7095	0.7010	0.7182	0.5187
+ExamInk-Evo(WorkBuddy)	0.8763	0.8635	0.8894	0.7746	0.8268	0.8147	0.8392	0.6919	0.7420	0.7312	0.7531	0.5759
Δ (%)	**+2.7%**	**+2.4%**	**+3.0%**	**+6.5%**	**+3.5%**	**+3.2%**	**+3.8%**	**+8.2%**	**+4.6%**	**+4.3%**	**+4.9%**	**+11.0%**
WorkBuddy (DeepSeek-V4.1-Flash)	0.7671	0.7738	0.7605	0.5940	0.6883	0.6942	0.6824	0.4937	0.5783	0.5833	0.5733	0.3710
+ExamInk-Evo(Codex)	0.8087	0.7943	0.8236	0.6527	0.7308	0.7178	0.7443	0.5557	0.6273	0.6161	0.6389	0.4316
Δ (%)	**+5.4%**	**+2.6%**	**+8.3%**	**+9.9%**	**+6.2%**	**+3.4%**	**+9.1%**	**+12.6%**	**+8.5%**	**+5.6%**	**+11.4%**	**+16.3%**

(b) Intra-line Transcription Metrics

System	$\tau_{\text{line}} = 0.4$			$\tau_{\text{line}} = 0.5$			$\tau_{\text{line}} = 0.6$		
	ROUGE-L↑	CER↓	TER↓	ROUGE-L↑	CER↓	TER↓	ROUGE-L↑	CER↓	TER↓
Codex (GPT-5.6-sol)	0.7425	0.3658	0.2950	0.7102	0.4281	0.3612	0.6501	0.5397	0.4816
+ExamInk-Evo(WorkBuddy)	0.7750	0.3288	0.2624	0.7456	0.3850	0.3221	0.6885	0.4905	0.4349
Δ (%)	**+4.4%**	**+10.1%**	**+11.1%**	**+5.0%**	**+10.1%**	**+10.8%**	**+5.9%**	**+9.1%**	**+9.7%**
WorkBuddy (DeepSeek-V4.1-Flash)	0.6428	0.6780	0.6128	0.5949	0.8069	0.7380	0.5185	0.9955	0.9268
+ExamInk-Evo(Codex)	0.7013	0.6247	0.5902	0.6525	0.7601	0.7163	0.5788	0.9337	0.8801
Δ (%)	**+9.1%**	**+7.9%**	**+3.7%**	**+9.7%**	**+5.8%**	**+2.9%**	**+11.6%**	**+6.2%**	**+5.0%**

6 CONCLUSION

We introduced **ExamInkBench**, a benchmark for end-to-end student answer reconstruction from complete examination pages, comprising 141 public samples drawn from a 4,017-sample fully annotated corpus selected from 31,769 examination papers. Its question-aware, two-level protocol evaluates line-level extraction and ordering together with intra-line textual and mathematical transcription. Evaluations of 11 mainstream multimodal systems show that answer-line coverage and structural reconstruction remain major bottlenecks, and that accurate local recognition does not guarantee correct page-level answers. We further proposed **ExamInk-Evo**, a failure-driven skill self-evolution framework that consistently improves extraction and transcription for tool-using agents, especially under stricter matching thresholds, while also enabling positive cross-agent skill transfer under different thresholds.

Limitations. Our benchmark currently focuses on mathematics examination pages. Future work will further investigate other subjects and examination formats.

AI USE STATEMENT

In this work, we used generative AI tools to aid and polish writing, including sentence rephrasing, grammar correction, and improving readability of manuscript text. Generative AI was also used for retrieval and discovery (e.g., finding related work). Additionally, no other generative AI-assisted tasks are involved. We have reviewed all AI-assisted content thoroughly. All outputs produced by generative AI were manually checked and revised by the authors to ensure accuracy, consistency and scientific rigor. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work introduces ExamInkBench, a benchmark built upon real-world mathematics examination papers. All source examination materials used for dataset construction were obtained with authorization from participating students and relevant organizations. Personally identifiable student information is completely removed during data curation. No raw handwritten samples containing real-name, student ID numbers or other private attributes are retained in the final released benchmark to protect student privacy.

We strictly comply with relevant data-use and copyright regulations for educational documents. The released benchmark is intended exclusively for non-commercial academic research purposes. We prohibit any misuse of this dataset for commercial grading services, surveillance or other applications outside educational research. No new human-subject experiments were conducted in this study, therefore IRB review was not required.

REPRODUCIBILITY STATEMENT

To support reproducibility of our work, we will release the public evaluation split of ExamInkBench upon paper acceptance. Detailed dataset curation, annotation pipeline, and evaluation protocol are described in Section 3 and Appendix B of the main paper. The implementation of our proposed ExamInk-Evo failure-driven skill self-evolution framework, including agent execution, failure mining, skill generation and validation logic, will also be made available upon paper acceptance. The experimental settings, objective function definitions and experimental configurations for both self-evolution and cross-agent transfer experiments are documented in Appendix D. All experimental metrics, including line-level extraction and intra-line transcription measurements, follow the formulas reported in the paper, enabling direct reproduction of our quantitative results.

REFERENCES

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

Shuaiqi Duan, Yadong Xue, Weihan Wang, Zhe Su, Huan Liu, Sheng Yang, Guobing Gan, Guo Wang, Zihan Wang, Shengdong Yan, et al. Glm-ocr technical report. *arXiv preprint arXiv:2603.10910*, 2026.

Lenardo Chaves e Silva, Álvaro Sobrinho, Thiago Cordeiro, Alan Pedro da Silva, Diego Dermeval, Leonardo Brandão Marques, Ig Ibert Bittencourt, Jário José dos Santos Júnior, Rafael Ferreira Melo, Carlos dos Santos Portela, et al. Assessing students' handwritten text productions: A two-decades literature review. *Expert Systems with Applications*, 250:123780, 2024.

Ling Fu, Zhebin Kuang, Jiajun Song, Mingxin Huang, Biao Yang, Yuzhe Li, Linghao Zhu, Qidi Luo, Xinyu Wang, Hao Lu, et al. Ocrbench v2: An improved benchmark for evaluating large multimodal models on visual text localization and reasoning. *Advances in Neural Information Processing Systems*, 38, 2026.

Philippe Gervais, Anastasiia Fadeeva, and Andrii Maksai. Mathwriting: A dataset for handwritten mathematical expression recognition. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 5459–5469, 2025.

Christian Gold and Torsten Zesch. Exploring the impact of handwriting recognition on the automated scoring of handwritten student answers. In *2020 17th international conference on frontiers in handwriting recognition (ICFHR)*, pp. 252–257. IEEE, 2020.

Emmanuèle Grosicki, Matthieu Carré, Jean-Marie Brodin, and Edouard Geoffrois. Results of the rimes evaluation campaign for handwritten mail processing. In *2009 10th international conference on document analysis and recognition*, pp. 941–945. IEEE, 2009.

Zinuo Guo, Min Zhang, and Bo Jiang. Omnihandwritingocr: A diagnostic benchmark for evaluating multimodal llms in handwritten ocr scenarios. *arXiv preprint arXiv:2608.18586*, 2026.

Liufeng Huang, Bangdong Chen, Chongyu Liu, Dezhi Peng, Weiying Zhou, Yaqiang Wu, Hui Li, Hao Ni, and Lianwen Jin. Ensexam: a dataset for handwritten text erasure on examination papers. In *International Conference on Document Analysis and Recognition*, pp. 470–485. Springer, 2023.

Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. Layoutlmv3: Pre-training for document ai with unified text and image masking. In *Proceedings of the 30th ACM international conference on multimedia*, pp. 4083–4091, 2022.

Guillaume Jaume, Hazim Kemal Ekenel, and Jean-Philippe Thiran. Funsd: A dataset for form understanding in noisy scanned documents. In *2019 international conference on document analysis and recognition workshops (ICDARW)*, volume 2, pp. 1–6. IEEE, 2019.

Zixuan Ke and Vincent Ng. Automated essay scoring: A survey of the state of the art. In *IJCAI*, volume 19, pp. 6300–6308, 2019.

Geewook Kim, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. Ocr-free document understanding transformer. In *European Conference on Computer Vision*, pp. 498–517. Springer, 2022.

Minghao Li, Tengchao Lv, Jingye Chen, Lei Cui, Yijuan Lu, Dinei Florencio, Cha Zhang, Zhoujun Li, and Furu Wei. Trocr: Transformer-based optical character recognition with pre-trained models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 13094–13102, 2023.

Runrui Li, Lin Zhu, and Hua Huang. Bridging the gap in exam handwritten text recognition: Dataset, benchmark, and modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pp. 74–81, 2004.

Yuliang Liu, Zhang Li, Mingxin Huang, Biao Yang, Wenwen Yu, Chunyuan Li, Xu-Cheng Yin, Cheng-Lin Liu, Lianwen Jin, and Xiang Bai. Ocrbench: on the hidden mystery of ocr in large multimodal models. *Science China Information Sciences*, 67(12):220102, 2024.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, and Jianfeng Gao. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. In *International Conference on Learning Representations*, volume 2024, pp. 23439–23554, 2024.

U-V Marti and Horst Bunke. The iam-database: an english sentence database for offline handwriting recognition. *International journal on document analysis and recognition*, 5(1):39–46, 2002.

Minesh Mathew, Dimosthenis Karatzas, and CV Jawahar. Docvqa: A dataset for vqa on document images. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 2199–2208. IEEE, 2021.

Linke Ouyang, Yuan Qu, Hongbin Zhou, Jiawei Zhu, Rui Zhang, Qunshu Lin, Bin Wang, Zhiyuan Zhao, Man Jiang, Xiaomeng Zhao, et al. Omnidocbench: Benchmarking diverse pdf document parsing with comprehensive annotations. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24838–24848. IEEE, 2025.

Birgit Pfitzmann, Christoph Auer, Michele Dolfi, Ahmed S Nassar, and Peter Staar. Doclaynet: A large human-annotated dataset for document-layout segmentation. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 3743–3751, 2022.

Qwen Team. Qwen3. 5-omni technical report. *arXiv preprint arXiv:2604.15804*, 2026.

Zilong Wang, Yiheng Xu, Lei Cui, Jingbo Shang, and Furu Wei. Layoutreader: Pre-training of text and layout for reading order detection. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 4735–4744, 2021.

Anyi Xu, B Li, Bangcai Lin, Bing Xue, BingCheng Xian, Bingzheng Xu, Bochao Wu, Bowei Zhang, Boyi Deng, CC Yu, et al. Deepseek-v4. 1-flash: Pushing the limits of kv cache compression. *arXiv preprint arXiv:2609.19969*, 2026.

Hesuo Zhang, Lingyu Liang, and Lianwen Jin. Scut-hccdoc: A new benchmark dataset of handwritten chinese text in unconstrained camera-captured documents. *Pattern Recognition*, 108:107559, 2020.

Chuangxin Zhao, Boyan Shi, Yanling Wang, Yijian LU, Canran Xiao, Jiali Chen, Jun Xia, Yan Wang, Ji Qi, and Juanzi Li. Hg-bench: A benchmark for multi-page handwritten answer-region grounding in automated homework assessment. *arXiv preprint arXiv:2606.25491*, 2026.

A Inter-Annotator Agreement

A.1 Agreement Measures

The public evaluation split contains 141 samples. Inter-annotator agreement is evaluated on 75 samples randomly sampled without replacement from this public evaluation split. For each selected sample, we use the eight original independent annotations collected before review or final adjudication.

For inter-annotator agreement analysis, we construct auxiliary image-grounded correspondence units from answer fragments, content regions, and internal line boundaries. These units are used only to align the pre-adjudication annotations, not as part of the benchmark schema or model evaluation protocol. They preserve question labels, line segmentation, ordering, and text, with correspondence determined directly from the examination images.

Question coverage. For annotators a and b, let $Q^{(a)}$ and $Q^{(b)}$ be their normalized question-identifier sets, including unanswered questions. Question-set F1 is

$$\frac{2|Q^{(a)} \cap Q^{(b)}|}{|Q^{(a)}| + |Q^{(b)}|}.$$

Answer association. Using the auxiliary image-grounded fragments defined above, answer-assignment agreement is the fraction of fragments assigned to the same question by both annotators, over the union of fragments recorded by either. A fragment recorded by only one annotator counts as disagreement. Question identifiers follow Appendix B.1.

Answer-line segmentation. For internal boundary sets $B^{(a)}$ and $B^{(b)}$, line-boundary F1 is $2|B^{(a)} \cap B^{(b)}|/(|B^{(a)}| + |B^{(b)}|)$. Boundaries are identified by the image-grounded fragments on their two sides, excluding outer answer-region boundaries. Splitting or merging lines changes this boundary set.

Answer-line ordering. Relative-order agreement is the fraction of common visual-line pairs with the same relative order. Common lines contain the same fragments in both annotations; eligible pairs belong to the same image-grounded answer, regardless of assigned question labels.

Text recognition. For normalized character sequences u and v from the same content region, agreement is $1 - \mathrm{Lev}(u, v)/\max(|u|, |v|)$, using Appendix B.1. Fragments are concatenated in the shared image-grounded order with fixed inter-fragment spacing, separating content agreement from line segmentation and ordering. A region missing from one annotation receives zero similarity.

A.2 Aggregation and Reporting

Each sample yields $\binom{8}{2} = 28$ annotator pairs. Per dimension, applicable pairwise scores are averaged within samples, then macro-averaged across eligible samples. Text similarity first averages regions within each pair. Comparisons without supporting identifiers, fragments, boundaries, or content regions are inapplicable; ordering requires a common line pair. Table A.1 summarizes the five agreement measures and their eligible sample counts.

Table A.1: Agreement reporting for the 75-sample subset. All scores lie in $[0, 1]$, and $n \leq 75$ is the eligible sample count.

Dimension	Measure	Observed	n
Question coverage	Question-set F1	0.982	75
Answer association	Answer-assignment agreement	0.957	75
Answer-line segmentation	Line-boundary F1	0.913	74
Answer-line ordering	Relative-order agreement	0.976	73
Text recognition	Normalized edit similarity	0.948	75

B Detailed Evaluation Protocol

B.1 Representation and Normalization

Questions are keyed by sample identifier and complete question identifier. Scoring covers the union of gold and predicted questions; absent questions and a sole question-level `None` record become empty answer sequences.

Output validation. JSON is decoded once after optionally removing one enclosing code fence. Each group has a unique `question_id` after identifier normalization and an ordered `lines` array of string `text` fields and consecutive `line_index` values starting at one. Invalid JSON, duplicate identifiers, invalid indices, non-string or normalized-empty lines, and `None` mixed with nonempty lines count as output failures. Such a sample receives an empty prediction, retaining its gold lines and a failure count. Gold structures are validated in advance; repeated lines retain separate occurrences.

Normalization. Question identifiers use Unicode NFC, trimmed whitespace, ASCII digits and parentheses in place of full-width forms, and no spaces adjacent to parentheses; subquestion hierarchy is preserved. For matching, CER, and ROUGE-L, the shared function ϕ applies NFC, converts tabs and within-line breaks to spaces, collapses whitespace to one ASCII space, and trims the result. Mathematical delimiters `\(...\)`, `\[...\]`, `$...$`, and `$$...$$` are canonicalized to `\(...\)`. Spans are scanned left to right without overlap, recognizing the longest opening delimiter first and its corresponding closer; escaped delimiters remain literal. Character scoring includes the canonical delimiters and LaTeX source, preserving case, operators, punctuation, structure, and student errors without semantic rewriting.

Mathematical tokens. From $\phi(g)$, $t(g)$ concatenates mathematical spans in order, excluding delimiters and adding no separators. Each LaTeX control word or symbol (including `\\`) is one token; other characters are individual tokens. Tokenization precedes removal of inter-token whitespace: `\sin x` becomes `\sin` and `x`. Spaces inside `\text{...}` are collapsed and retained; other mathematical whitespace is ignored. Commands and braces are retained without expansion. An unclosed opening delimiter extends to the line end and is flagged; content without an opening delimiter remains in character scoring only.

B.2 Question-aware Matching and Metric Aggregation

Let $\mathcal{Q}$ be the union of gold and predicted question identifiers across the evaluated split, with each identifier including sample identity. For $q \in \mathcal{Q}$, G_q and P_q contain n_q gold and m_q predicted lines; an absent question has an empty sequence. The split-level line counts are $N_{\text{gt}} = \sum_q n_q$ and $N_{\text{pred}} = \sum_q m_q$. The normalization and tokenization functions follow Appendix B.1.

Question-aware line matching. For $1 \le i \le n_q$ and $1 \le j \le m_q$, normalized edit similarity is

$$s_{ij}^{(q)} = 1 - \frac{\mathrm{Lev}(\phi(g_{q,i}), \phi(p_{q,j}))}{\max(|\phi(g_{q,i})|, |\phi(p_{q,j})|)}. \tag{B.1}$$

Here Lev counts unit-cost character substitutions, deletions, and insertions. Let $\mathcal{A}_q$ contain all assignments of $\min(n_q, m_q)$ index pairs, with each gold and predicted index used at most once. Hungarian matching selects

$$A_q = \arg\max_{A \in \mathcal{A}_q} \sum_{(i,j) \in A} s_{ij}^{(q)}. \tag{B.2}$$

Threshold filtering then defines the valid matches:

$$M_q = \left\{ (i,j) \in A_q : s_{ij}^{(q)} \ge \tau_{\text{line}} \right\}. \tag{B.3}$$

If either sequence is empty, $A_q = M_q = \varnothing$. Validation excludes normalized-empty ordinary lines. Filtering does not trigger reassignment; equally optimal assignments use the lexicographically smallest sorted list of index pairs. The matching objective uses content alone, with ordering evaluated separately. Benchmark results are reported separately at $\tau_{\text{line}} \in \{0.4, 0.5, 0.6\}$.

Line extraction. Let $K_q = |M_q|$ be the number of valid matches for question q. Unmatched predicted and gold lines count as false positives and false negatives, respectively, including lines under additional or missing question identifiers. Counts are accumulated over the evaluated split before computing the micro-averaged scores:

$$P_{\text{line}} = \frac{\sum_q K_q}{\sum_q m_q}, \qquad R_{\text{line}} = \frac{\sum_q K_q}{\sum_q n_q}, \qquad F1_{\text{line}} = \frac{2\sum_q K_q}{\sum_q (m_q + n_q)}. \tag{B.4}$$

Higher precision indicates a larger fraction of valid extracted lines, higher recall indicates more complete line coverage, and higher F1 reflects better overall line extraction.

Line ordering. For valid matches $(i,j), (i',j') \in M_q$ with $i < i'$, the order is correct when $j < j'$. Let C_q count correctly ordered pairs and T_q count all comparable pairs:

$$\begin{gathered} C_q = \sum_{\substack{(i,j),(i',j') \in M_q \\ i < i'}} \mathbf{1}[j < j'], \qquad T_q = \binom{K_q}{2}, \\ \mathrm{LOA} = \frac{\sum_q C_q}{\sum_q T_q}. \end{gathered} \tag{B.5}$$

Questions with fewer than two valid matches contribute no pairs. Pooling comparable pairs gives higher weight to questions with more such pairs. Higher LOA indicates better preservation of the relative order of matched answer lines.

Character error rate. Let $\mathcal{M} = \{(q,i,j) : q \in \mathcal{Q},\ (i,j) \in M_q\}$ collect valid matched records. Let $\mathcal{U}_G$ and $\mathcal{U}_P$ contain all unmatched gold records (q,i) and predicted records (q,j), respectively. These sets include lines under missing or additional question identifiers. For $r = (q,i,j) \in \mathcal{M}$, define $u_r = \phi(g_{q,i})$ and $v_r = \phi(p_{q,j})$; let $S_{C,r}$, $D_{C,r}$, and $I_{C,r}$ denote character substitutions, deletions, and insertions in their minimum-edit alignment. Unmatched gold lines contribute complete-text deletions, and unmatched predicted lines contribute complete-text insertions:

$$D_C^U = \sum_{(q,i) \in \mathcal{U}_G} |\phi(g_{q,i})|, \qquad I_C^U = \sum_{(q,j) \in \mathcal{U}_P} |\phi(p_{q,j})|. \tag{B.6}$$

The denominator counts normalized characters in all gold lines:

$$N_{\text{char}}^{\text{GT}} = \sum_{q \in \mathcal{Q}} \sum_{i=1}^{n_q} |\phi(g_{q,i})|. \tag{B.7}$$

To make explicit the aggregate edit counts used in Equation 6 of the main text, we decompose them into matched-line and unmatched-line contributions. CER therefore pools the matched-line edits and unmatched-line contributions as

$$\mathrm{CER} = \frac{\sum_{r\in\mathcal{M}}(S_{C,r} + D_{C,r} + I_{C,r}) + D_C^U + I_C^U}{N_{\mathrm{char}}^{\mathrm{GT}}}. \tag{B.8}$$

Specifically, the aggregate quantities in the main text are

$$S_C = \sum_{r\in\mathcal{M}} S_{C,r}, \qquad D_C = \sum_{r\in\mathcal{M}} D_{C,r} + D_C^U, \qquad I_C = \sum_{r\in\mathcal{M}} I_{C,r} + I_C^U. \tag{B.9}$$

Thus, Equation B.8 is exactly equivalent to the compact definition $\mathrm{CER} = (S_C + D_C + I_C)/N_{\mathrm{char}}^{\mathrm{GT}}$ in Equation 6 of the main text. CER is reported without clipping. Lower CER indicates fewer character errors per gold character, accounting for both transcription errors and unmatched lines.

ROUGE-L F1. For each matched record, let $L_r = \mathrm{LCS}(u_r, v_r)$ be the longest-common-subsequence length of the normalized character sequences. Each unmatched gold line receives a score of zero. The matched-line F1 and the average over all gold lines are

$$F1_r^{\mathrm{LCS}} = \frac{2L_r}{|u_r| + |v_r|}, \qquad \text{ROUGE-L} = \frac{1}{N_{\mathrm{gt}}} \sum_{r\in\mathcal{M}} F1_r^{\mathrm{LCS}}. \tag{B.10}$$

Every gold line receives equal weight (Lin, 2004). Higher ROUGE-L F1 indicates more complete recovery of gold-line content.

LaTeX token error rate. For $r = (q, i, j) \in \mathcal{M}$, use the token sequences $u_r^T = t(g_{q,i})$ and $v_r^T = t(p_{q,j})$. Let $S_{T,r}$, $D_{T,r}$, and $I_{T,r}$ denote substitutions, deletions, and insertions in their minimum-edit alignment. All tokens in unmatched gold lines count as deletions, and all tokens in unmatched predicted lines count as insertions:

$$D_T^U = \sum_{(q,i)\in\mathcal{U}_G} |t(g_{q,i})|, \qquad I_T^U = \sum_{(q,j)\in\mathcal{U}_P} |t(p_{q,j})|. \tag{B.11}$$

The denominator counts mathematical tokens in all gold lines:

$$N_{\mathrm{token}}^{\mathrm{GT}} = \sum_{q\in\mathcal{Q}} \sum_{i=1}^{n_q} |t(g_{q,i})|. \tag{B.12}$$

To make explicit the aggregate edit counts used in Equation 7 of the main text, we decompose them into matched-line and unmatched-line contributions. The split-level LaTeX token error rate is therefore

$$\mathrm{TER}_{\mathrm{LaTeX}} = \frac{\sum_{r\in\mathcal{M}}(S_{T,r} + D_{T,r} + I_{T,r}) + D_T^U + I_T^U}{N_{\mathrm{token}}^{\mathrm{GT}}}. \tag{B.13}$$

Specifically, the aggregate quantities in the main text are

$$S_T = \sum_{r\in\mathcal{M}} S_{T,r}, \qquad D_T = \sum_{r\in\mathcal{M}} D_{T,r} + D_T^U, \qquad I_T = \sum_{r\in\mathcal{M}} I_{T,r} + I_T^U. \tag{B.14}$$

Thus, Equation B.13 is exactly equivalent to the compact definition $\mathrm{TER}_{\mathrm{LaTeX}} = (S_T + D_T + I_T)/N_{\mathrm{token}}^{\mathrm{GT}}$ in Equation 7 of the main text. Predicted tokens on a matched line without gold mathematical tokens count as insertions; two empty token sequences contribute zero edits. Token error rate is reported without clipping. Lower token error rate indicates fewer mathematical-token errors per gold token, accounting for both transcription errors and unmatched lines.

B.3 Boundary Conventions

Two empty answer sequences contribute no line counts. If $N_{\text{gt}} > 0$ but $N_{\text{pred}} = 0$, all extraction scores are zero. If $N_{\text{gt}} = 0 < N_{\text{pred}}$, precision and F1 are zero and recall is N/A; when both totals are zero, all three are N/A. CER is N/A only when $N_{\text{char}}^{\text{GT}} = 0$, ROUGE-L is N/A only when $N_{\text{gt}} = 0$, and token error rate is N/A only when $N_{\text{token}}^{\text{GT}} = 0$. Predicted-character and predicted-token insertions are recorded even when the corresponding denominator is zero. If gold lines are present but no valid matches exist, ROUGE-L is zero; all gold content contributes deletions and all predicted content contributes insertions to CER and token error rate. With gold lines present, an empty prediction has CER equal to one and, when gold mathematical tokens are present, token error rate equal to one. LOA is N/A when $\sum_q \binom{K_q}{2} = 0$.

For the development objective in Equation 11, after the clipping operation defined in the main text, an undefined component contributes zero utility, including the entire $1 - \overline{\text{CER}}$ or $1 - \overline{\text{TER}}_{\text{LaTeX}}$ term; the denominator remains five. Metric reports retain N/A. The evaluator records its configuration and software version, output failures, line totals, valid matches, correctly ordered and comparable pairs, total gold character and token counts, and the edit counts from matched and unmatched lines. Cross-question diagnostics in Section 5.2 leave benchmark matches unchanged.

C MLLM Benchmark Evaluation Protocol

To ensure rigorous reproducibility and prevent data contamination, all evaluated Multimodal Large Language Models (MLLMs) are benchmarked under a uniform and deterministic execution protocol.

C.1 Task Definition and Reference Benchmark Setup

- **Evaluation Dataset:** Models are evaluated on the complete public evaluation split of 141 camera-captured and scanned mathematics examination papers.
- **Deterministic Evaluator:** Model predictions are scored against gold annotations using the deterministic evaluator script `src/evaluator.py` under the fixed matching threshold $\tau_{\text{line}} = 0.4/0.5/0.6$.
- **Reconstruction Goal:** Given an exam-page image and valid question identifiers, the MLLM reconstructs student answer lines, groups them by question, preserves their writing order, and outputs a structured JSON object.

C.2 Standardized Prompt Template

Every model is queried using the same single-turn prompt template. The placeholder {`question_ids`} is populated using the exact serialization call `json.dumps(question_ids, ensure_ascii=False)`.

The input follows the standard multimodal message format:

```
messages = [
    {
        "role": "user",
        "content": [
            {
                "type": "image_url",
                "image_url": {
                    "url": "..."
                }
            },
            {
                "type": "text",
                "text": prompt
            }
        ]
    }
]
```

The verbatim prompt is shown below:

```
You are an expert handwritten mathematics examination parser.
Your task is to transcribe all student-written answer lines from the provided examination
     paper image, grouping them strictly by question identifier and preserving the visible
      writing order.

Task Instructions:
1. Candidate Question Identifiers:
The valid question items to associate answers with are:
{question_ids}

2. Extraction Rules:
- Reconstruct student handwriting on an atomic line-by-line basis.
- Exclude printed questions, printed headers, scratch markings, crossed-out lines, and
     teacher scoring annotations.
- Transcribe mathematical formulas, symbols, and expressions into clean standard LaTeX
     notation.
- Preserve the authentic relative reading and writing order of lines within each question.
- If a question identifier listed above has NO handwritten student response on the page,
     return a single line with text "None".

3. Output Format:
Return ONLY a valid JSON object matching the following structure without any
     conversational text or markdown code fences:
{
  "answers": [
    {
      "question_id": "1",
      "lines": [
        {"line_index": 1, "text": "line content 1"},
        {"line_index": 2, "text": "line content 2"}
      ]
    },
    {
      "question_id": "2",
      "lines": [
        {"line_index": 1, "text": "None"}
      ]
    }
  ]
}
```

C.3 Inference Protocol and Hard Execution Constraints

All evaluation runs follow the same operational constraints:

1. **Single-Pass Inference:** Exactly one inference request is executed per sample. Multi-turn correction, critique, or re-prompting is forbidden.
2. **Deterministic Decoding:** Generation randomness is minimized using temperature $= 0$ or the provider's minimum permitted value. Runtime parameters are logged.
3. **Deterministic Image Preprocessing:** Images are processed with PIL, auto-oriented using EXIF metadata, resized to a maximum dimension of 2000 px, and saved as JPEG with quality 85.
4. **Ground-Truth Isolation:** The inference pipeline cannot access gold annotations or existing result files.
5. **Atomic Persistence:** Each prediction is written immediately after its sample is processed. Existing completed outputs are skipped when resuming.
6. **Question ID Injection:** Candidate `question_ids` are copied from the task definition. Questions without visible handwriting receive the exact output string `"None"`.

C.4 Output Serialization Schema

Each sample produces an independent JSON file at:

```
runs/benchmark_mllm/<model_tag>/preds/<sample_id>.json
```

D Implementation Details of Failure-Driven Skill Self-Evolution

In this section, we elaborate on the implementation specifics of ExamInk-Evo, detailing the programmatic execution model, evaluation objective, evolutionary hyperparameters, role prompt specifications, and the task-specific skill catalog.

D.1 Evaluation Criteria and Selection Objective

Candidate agent programs are evaluated and selected using a deterministic evaluation engine (`src/evaluator.py`) under a fixed line-matching threshold $\tau_{\text{line}} = 0.5$.

Matching and Aggregation. Valid line-level correspondence requires a normalized edit similarity $s_{ij}^{(q)} \geq 0.5$ via independent Hungarian matching per question. To account for both coverage and transcription quality while preventing divergent error penalties from dominating selection, error metrics are bounded by unit clipping:

$$\overline{\text{CER}} = \min(\text{CER}, 1), \quad \overline{\text{TER}}_{\text{LaTeX}} = \min(\text{TER}_{\text{LaTeX}}, 1). \tag{D.1}$$

The evolutionary selection score $\mathcal{S}_{\text{dev}}(p)$ is computed as an unweighted average of the five primary normalized dimensions:

$$\mathcal{S}_{\text{dev}}(p) = \frac{1}{5}\left[\text{LineF1} + \text{LOA} + \text{ROUGE-L} + (1 - \overline{\text{CER}}) + (1 - \overline{\text{TER}}_{\text{LaTeX}})\right]. \tag{D.2}$$

The matching threshold $\tau_{\text{line}} = 0.5$ serves as the sole canonical criterion during optimization; auxiliary threshold evaluations ($\tau_{\text{line}} \in \{0.4, 0.6\}$) are used exclusively to assess post-hoc robustness.

D.2 Evolution Hyperparameters and Search Discipline

The iterative optimization loop maintains a bounded Pareto frontier governed by strictly deterministic update rules (summarized in Table D.1).

Table D.1: Hyperparameter configuration of the ExamInk-Evo optimization loop.

Category	Hyperparameter	Value / Specification
Frontier	Capacity (k)	3 candidate programs
	Parent Selection	Round-robin across active frontier programs
	Replacement Rule	Strictly better ($\mathcal{S}_{\text{candidate}} > \min_{p \in \mathcal{G}} \mathcal{S}_{\text{dev}}(p)$; ties rejected)
Optimization	Max Iterations (T)	10
	Early Stopping	Terminate after 3 consecutive non-improving iterations
	Mutation Budget	Exactly 1 mutation (`CREATE` or `EDIT`) per iteration
Failure Mining	Strategy	Recurrent pattern clustering
	Primary Ranking	Frequency-first
	Tie-Breaking Criterion	Aggregate cumulative error magnitude
	Exemplar Capacity	16 representative failure traces
Logging	Observable Traces	Full state serialization (prompts, skill diffs, execution traces)

Deterministic Application Engine. To decouple stochastic API throttling and inference overhead from the evolutionary loop, baseline perception is grounded once into disk-backed traces. Running a candidate program p consists of sequentially applying its structured skill transformations via a deterministic realization engine $\text{REALIZE}(p)$. When addressing coverage bottlenecks, visual re-perception acts as a perceptual replacement operator that supersedes erroneous base transcriptions through cached line-level re-extractions, rather than attempting text-only hallucination.

D.3 Role Prompts and Operational Protocols

The self-evolution loop instantiates three specialized roles: *Executor*, *Proposer*, and *Skill Builder*.

Proposer Mapping. At iteration t, recurrent failure diagnostics $\mathcal{D}(x; p)$ are clustered across six structural dimensions: *coverage*, *text_transcription*, *math_transcription*, *association*, *ordering*, and *question_discovery*. To guarantee reproducibility and eliminate meta-prompt variance, the Proposer utilizes a deterministic pattern-to-skill dispatch table:

- `coverage` → `line-reperception`
- `text_transcription` → `text-normalization`
- `math_transcription` → `math-latex-normalization`
- `association` → `question-id-normalization`
- `ordering` → `line-order-stabilization`
- `question_discovery` → `none-line-protocol`

If the designated skill is absent from the parent repository, the Proposer emits a `CREATE` action; otherwise, it triggers an `EDIT` action to refine active operational parameters (e.g., enabling sub-phrase delimiter splitting or Unicode canonicalization).

Skill Builder. The Skill Builder materializes the proposed specification into a modular skill descriptor (`SKILL.md`) equipped with YAML frontmatter specifying execution triggers, target dimensions, and parameter overrides, producing an explicit `unified diff` against the parent repository.

Line-Focused Visual Re-perception Prompt. For perceptual skill execution, secondary visual sub-agents are driven by a line-focused transcription prompt with strict anti-leakage isolation:

```
You are an expert handwritten mathematics transcriber tasked with exhaustive, line-by-line
      recovery.

Operational Constraints:
1. Question Anchor Integrity:
Strictly reproduce the candidate question items provided in the job specification: {
     question_ids}. Do not omit, reorder, or introduce arbitrary question identifiers.

2. Atomic Line Reconstruction:
- Transcribe handwriting on a physical line-by-line basis, strictly preserving the top-to-
     bottom page order.
- Transcribe mathematical formulas into standard inline LaTeX \( ... \).
- Preserve student errors, incorrect derivations, and cancellations verbatim; do not infer
      or mathematically self-correct.
- For question items containing no student handwriting, return a single line with text "
     None". The text field must not be empty.

3. Isolation Protocol:
Do not access external caches or reference label files. Return raw, well-formed JSON
     conforming to the target schema.
```

D.4 Task Skill Catalog

The atomic skill repository ($\mathcal{S}_{\text{catalog}}$) consists of eight deterministic transformation operators:

1. **`answer-line-harvest`** (*Coverage*): Normalizes segmented lines into atomic visual rows based on punctuation and line-break delimiters. Optional argument: `split_on_comma=true`.
2. **`text-normalization`** (*Text Transcription*): Performs Unicode whitespace folding, removes non-printable control characters, and standardizes full-width parentheses.
3. **`math-latex-normalization`** (*Math Transcription*): Standardizes LaTeX math environments into a canonical representation:

- *Macro Stripping:* Strips purely presentational directives including `\left`, `\right`, `\!`, `\,`, `\quad`, `\displaystyle`, and alignment spaces.
- *Operator Canonicalization:* Normalizes fractional and inequality aliases (`\dfrac`, `\tfrac` → `\frac`; `\geq` → `\ge`; `\neq` → `\ne`; `\longrightarrow` → `\to`).
- *Symbol Mapping:* Maps literal Unicode symbols to standard LaTeX macros ($\times$ → `\times`, $\leq$ → `\le`, π → `\pi`, $\pm$ → `\pm`, ∂ → `\partial`).
- *Whitespace Convergence:* Eliminates redundant spaces within mathematical delimiters `\( ... \)` and surrounding relational operators ($=, +, -, <, >$).

4. **`question-id-normalization`** (*Association*): Strips formatting noise from question identifiers and normalizes full-width brackets to standard ASCII.
5. **`line-order-stabilization`** (*Ordering*): Sorts extracted lines strictly by verified vertical layout index, ensuring strictly monotonic index values.
6. **`none-line-protocol`** (*Question Discovery*): Explicitly inserts a sentinel `"None"` line for unanswered questions in the candidate set.
7. **`repetition-dedup`** (*Text Transcription*): Detects and collapses adjacent identical OCR hallucinations caused by repetitive decoding loops.
8. **`line-reperception`** (*Coverage*): Invokes the visual backend to replace the base sample predictions with high-resolution line-focused re-transcriptions, overcoming the recall plateau of text-only transformations.