

Can LLMs Model Incorrect Student Reasoning? A Case Study on Distractor Generation for Multiple-Choice Questions

Yanick Zengaffinen¹ Andreas Opedal¹ Donya Rooein² Kv Aditya Srivatsa¹
Shashank Sonkar³ Mrinmaya Sachan¹

¹ETH Zürich ²Bocconi University ³University of Central Florida
yanick.zengaffinen@inf.ethz.ch

Abstract

Modeling student misconceptions in a realistic manner is critical for AI in education. In this work, we examine how large language models (LLMs) reason about misconceptions when generating distractor answers for multiple-choice questions (MCQs), a task that requires producing answers that are incorrect, yet plausible. We introduce a taxonomy over reasoning strategies for distractor generation that is grounded in learning-science literature and empirical observation, which we apply to LLM-generated reasoning traces across math and science MCQs. On the math dataset, we find that models follow a misconception-based process with potentially high diagnostic value: they recover the correct solution, articulate student errors, simulate them, and select plausible candidates. On the science dataset, on the other hand, they tend to follow a less robust approach based on semantic similarity to the correct answer. We find the most frequent failure modes to be that the model is unable to generate a correct solution or that it discards plausible distractor candidates when performing selection. Providing the correct solution in the prompt yields a relative improvement of 6.4% in alignment with human-authored distractors, highlighting the critical role of anchoring distractor generation to the correct solution. Together, our findings offer an interpretable view of how LLMs model incorrect student reasoning.¹

1 Introduction

Student modeling is a foundational problem in AI for education (Matsuda et al., 2007; Käser and Alexandron, 2024), underpinning a range of applications such as test design (Benedetto et al., 2024; Ezzaki et al., 2025), misconception diagnosis, teacher training (Judge et al., 2013; Lee et al., 2023) and development of targeted instructional interventions (Jin et al., 2025; Rooein et al., 2026). A

central component of student modeling is the ability to represent not only what students know, but also the systematic errors and misconceptions that characterize incorrect but plausible reasoning. Despite rapid progress in large language model (LLM) capabilities and a growing focus on using LLMs for student simulation (Macina et al., 2023; Opedal et al., 2024; Liu et al., 2025; Wu et al., 2025), existing work has largely studied LLM alignment with student behavior by evaluating whether they arrive at similar final answers, leaving open the complementary question of *how* LLMs actually simulate student mistakes.

To answer this question, we study distractor generation, the task of generating incorrect but plausible answer alternatives, i.e., **distractors**, for multiple-choice questions (MCQs). Distractor generation is a natural application of student modeling: if a model can faithfully simulate plausible student errors, distractors can be obtained by sampling from simulated incorrect solutions and selecting probable outcomes. To produce good distractors, a model is required not only to identify the correct solution, but also to simulate *incorrect* student reasoning and to verify the plausibility of the resulting answers (Gierl et al., 2017; Haladyna and Rodriguez, 2013). Distractor generation is therefore a strictly more difficult task than the one of *solving* problems. While prior work has investigated using LLMs for distractor generation (Feng et al., 2024; Alhazmi et al., 2024; Doughty et al., 2024; Säuberli and Clematide, 2024; Parikh et al., 2025), the focus has largely been on evaluating final answers. These alone do not reveal whether distractors arise from pedagogically meaningful reasoning, or something more akin to surface-level heuristics. In educational settings, it is particularly important to understand the underlying process, both for aligning LLM strategies with expert practices and for building teacher trust in AI-based tools (Nazaretsky et al., 2022; Feldman-Maggor et al., 2025).

¹All code is available on GitHub at <https://github.com/eth-lre/llm-student-modeling-strategies>.

Task Interpretation	We are given a question: "Simplify the following if possible: $(6+9m)/3$ ". We need to generate TWO incorrect distractor answers.
Correct Answer Reference	The correct simplification is: $(6+9m)/3 = 3(2+3m)/3 = 2+3m$
High-Level Error Description	Possible Errors: (1) Forgetting to divide both terms by the denominator.
Error Simulation	e.g., only divide 6 by 3 but not 9m. This gets us $(6/3 + 9m) =$
Incorrect Outcome Instantiation	$2 + 9m$.
Incorrect Outcome Instantiation	or $6 + 3m$
Error Simulation	if 3 divides only 9m and not 6.
Plausibility Assessment	Both count towards the same error, so they are equally likely.
High-Level Error Description	Next, (2) They might try to combine the constant and variable.
Error Simulation	So, they might add 6 and 9m to get 15m, simplifying the expression to $15m/3=$
Incorrect Outcome Instantiation	$5m$.
Discriminability Check	This is clearly different from the correct answer
Final Set Curation	This gives us two distractors, one from each error type.
Reconsideration	But lets think again, are there other errors we should consider? No.

Figure 1: An LLM reasoning trace for distractor generation, annotated according to our taxonomy (Table 1). Color-coded labels show the model interpreting the task, anchoring to the correct solution, generating candidate distractors based on errors, evaluating their plausibility, and curating the final set. This mirrors strategies identified in the learning-science literature (§3.1). The LINK strategy, which captures conceptual similarity to the correct answer, does not appear in this trace, as it is rarely used on the math dataset (§4.2). In subsequent traces, we request three distractors rather than the two shown here.

We propose a framework for analyzing the reasoning traces generated by LLMs during distractor generation.² Each such reasoning **trace** corresponds to a single LLM generating the distractors for one MCQ. We construct a learning-science-aligned taxonomy of ten distractor-generation strategies, defined in Table 1 and grounded in principles of misconception-based assessment design. We complement the taxonomy with empirical observation through iterative open coding of LLM reasoning traces (Corbin and Strauss, 2014), a qualitative process for identifying and labeling recurring concepts in data. The taxonomy provides a theory-grounded lens for systematically characterizing and diagnosing *how* LLMs generate distractors, rather than only evaluating the final distractor set. We apply this taxonomy to 885 distractor-generation traces from DEEPSEEK-V3.2 (DeepSeek-AI, 2025) and GLM-4.7 (GLM-4.5 Team, 2025) across two MCQ datasets spanning mathematics and science. Fig. 1 illustrates the annotation of a single reasoning trace according to the strategies in our taxonomy.

Most notably, we find that on the misconception-annotated math dataset, LLM reasoning traces

show close alignment with best practices for distractor design (Haladyna and Rodriguez, 2013; Gierl et al., 2017): models recover the correct solution, articulate plausible student misconceptions, instantiate them through erroneous reasoning, and select plausible incorrect answers from a broader candidate set. The recovered solution serves as an anchor from which the model diverges at specific points to generate misconception-aligned distractors. On the science dataset, however, models rely more heavily on superficial semantic similarity to the correct answer: they recover the correct answer, establish a conceptual link to it, generate candidates along that link, and verify that these are indeed incorrect before selecting a final set. Across all settings, we observe a consistent high-level structure for problem understanding, candidate distractor construction, evaluation, and selection.

Beyond characterizing LLMs’ processes, our taxonomy also enables a diagnosis of where and why distractor generation fails. We find that some parts rarely fail: both error simulation, i.e., simulating a student’s flawed reasoning step by step, and the implicit organization of multi-step reasoning, i.e., the order in which strategies are chained. Note that the latter emerges without instruction: the model self-structures it. Instead, failures arise more often in recovering the correct solution and in the downstream selection of the final distractor set. Pro-

²Note that we do not study reconstruction of any real student’s reasoning; accordingly, we assess the reasoning traces against established pedagogical theory (misconception-based item design) rather than cognitive data from real students.

viding the correct solution in the prompt yields significant relative improvements in the fraction of human-authored distractors the model recovers (proportional match) of 6.4% on the math dataset and 30.7% on the science dataset,³ confirming that unreliable solution recovery is a primary bottleneck. We also find that models discard plausible candidates during selection, indicating that candidate ranking is another source of performance degradation.

Overall, our results show that LLMs can systematically engage in misconception-based distractor generation while adapting their strategies across domains, with correct-solution recovery serving as a key anchor. Together, these findings deepen our understanding of when and how modern LLMs can model plausible incorrect student reasoning.

2 Related Work

Student modeling with LLMs. A growing body of work suggests LLMs as promising tools for simulating student behavior. They can generate student responses that expert annotators judge as reflective of common grade-appropriate misconceptions (Macina et al., 2023) and condition their behavior on a learner’s cognitive level (Wu et al., 2025) or misconceptions (Liu et al., 2023). At the same time, recent literature suggests that LLMs may encode common student error patterns (Opedal et al., 2024; Liu et al., 2025), but it remains unclear whether they can effectively leverage this implicit knowledge for student simulation. While these findings suggest that LLMs can simulate incorrect student reasoning to some extent, prior evaluations focus primarily on final outputs, rather than the reasoning processes that produce them. As a result, it remains unclear which reasoning strategies LLMs use to generate incorrect answers.

Multiple-choice questions and distractor generation. MCQs are widely used to assess student understanding (Scouller, 1998; Chingos, 2012), but their effectiveness depends on the quality of the distractors. Crafting high-quality distractors (see §3.1) is costly and time consuming (Rudner, 2009; Quaigrain and Arhin, 2017). Prior work has therefore explored automating this process with

LLMs (Alhazmi et al., 2024, *inter alia*), particularly for mathematical MCQs (Feng et al., 2024; Hang et al., 2024; Lee et al., 2024; Parikh et al., 2025). Some approaches decompose distractor generation into producing candidate mistakes and applying them to generate plausible answers, reflecting the misconception-based approach formalized in §3.1 (Fernandez et al., 2024; Ross and Andreas, 2025), while others rely on overgeneration followed by ranking (Scarlatos et al., 2024). Despite notable performance gains, because this line of work measures only the final output set, the strategies underlying LLM-based distractor generation and their relation to learning-science principles remain insufficiently understood. We therefore complement it with a process-level analysis of the reasoning traces themselves, localizing failures to specific stages of the generation process.

3 A Taxonomy of Strategies for Distractor Generation

Our goal is to characterize the strategies LLMs use to generate distractors. We develop a taxonomy, shown in Table 1, by combining learning-science theory with empirical analysis to capture both expected strategies and observed LLM behavior. We first review experts’ strategies (§3.1), then describe how we construct our taxonomy (§3.2), and finally present the annotation pipeline used to apply it to LLM traces (§3.3).

3.1 Crafting Effective Distractors

Effective distractors must satisfy multiple criteria: they must be plausible, i.e., likely to be selected by students with flawed or incomplete understanding, be unambiguously incorrect, and not contain cues such as grammatical inconsistencies. In addition, a *set* of distractors must be mutually exclusive and avoid structural artifacts that reveal the correct answer (Haladyna and Rodriguez, 2013; Gierl et al., 2017; Tarrant et al., 2009; Haladyna et al., 2002). Learning-science literature distinguishes two main construction pipelines. **Similarity-based** distractors are generated by perturbing the surface form or semantic properties of the correct answer without modeling student reasoning (Chamberlain and Jeter, 2020). For example, if the correct answer is $\frac{4}{3}$, a distractor could invert the sign to $-\frac{4}{3}$. **Misconception-based** distractors, on the other hand, are derived from common student errors injected into an otherwise correct solution

³The larger relative gain on the science dataset should be interpreted with caution given the absolute improvement ($0.14 \rightarrow 0.18$) is small relative to the confidence intervals (± 0.02 and 0.03). Both improvements are significant under a paired t -test, $p < 0.01$.

process (Brown and VanLehn, 1980; VanLehn, 1990; Shin et al., 2019). For example, if students forget to invert the divisor when dividing fractions, a distractor can be generated by applying such an error (e.g., $\frac{1}{3} \div \frac{1}{2} \stackrel{!}{=} \frac{1}{6}$). Misconception-based distractors are more costly and complex to construct than similarity-based ones (Gierl et al., 2017), but are often preferred due to higher diagnostic value, as they expose specific procedural gaps in students’ understanding (Ashlock, 2010; De La Torre, 2009).

3.2 Taxonomy Construction

To develop a taxonomy that integrates expectations derived from expert practices (§3.1) with patterns observed in LLM distractor-generation traces, we first identify expected strategies from the literature, then extract strategies from actual LLM traces, and finally merge the two sets into a unified taxonomy.

Grounding in theory. From the approaches described above, we derive the following strategies, assigning each a tag (in parentheses) for reference throughout the paper. Both similarity-based and misconception-based approaches require access to the correct solution (CORR) and eventually instantiate concrete distractor values (INST). While similarity-based methods require establishing a semantic or conceptual link to the correct answer (LINK), misconception-based distractor generation instead involves identifying a common error (ERR_DESC) and simulating this error within an otherwise correct step-by-step solution procedure (ERR_SIM). Item-level quality constraints correspond to plausibility and discriminability checks (PLAUS, DISCR), while set-level constraints require explicit selection or curation over candidate distractors (CURATE).

Grounding in observation. To ensure our taxonomy represents actual, probable LLM reasoning, we collect and analyze reasoning traces from multiple models and datasets spanning different domains. We use a prompt that avoids biasing models toward any particular strategy; see Appendix §A.3. Each trace corresponds to one LLM generating distractors for one MCQ. To identify the strategies in LLMs’ reasoning, we perform manual open coding (Corbin and Strauss, 2014), a qualitative method for breaking text into discrete concepts without predefined categories, on a stratified subset of 48 traces across datasets, models, and levels of alignment with human-annotated distractors. To

further broaden coverage, we prompt DEEPSEEK-V3.2 to perform open coding on groups of 10 traces and report recurring strategies across each group. This is applied to 800 additional traces, without providing any previously extracted strategies. Experimental details are provided in §4.

Merging theory and observation. Finally, we manually aggregate and reconcile concepts derived from the literature, manual coding, and model-assisted coding, by aligning and merging overlapping and redundant concepts into broader strategies.⁴ This yields the taxonomy in Table 1. All ten strategies in our taxonomy were initially identified via manual open coding. LLM-assisted coding independently recovered all strategies except RECON and proposed additional codes that were excluded post hoc, as they captured problem-specific rather than general strategies. Eight of the ten final strategies are directly supported by distractor-generation literature (see Appendix §A.4). The remaining two (INTER and RECON) reflect general reasoning behaviors: INTER reflects task interpretation under structured item constraints, and RECON reflects iterative backtracking during solution refinement (Mislevy et al., 2003). Further details on strategy reconciliation are provided in Appendix §A.4.

3.3 Trace Annotation

To characterize the distribution of strategies LLMs use during distractor generation, we scale up annotation by applying the taxonomy to all traces, marking each span with a strategy tag as illustrated in Fig. 1. Following prior work, we use an LLM as the primary annotator and validate the resulting annotations manually on a representative subset of traces (Marjanovic et al., 2026), comparing insertions and deletions against human-annotated references. The LLM is prompted with (i) a description of the taxonomy, (ii) a small set of hand-picked examples per strategy tag grounded in observations from manual open coding, (iii) one full trace, and (iv) an instruction to re-generate the trace with strategy tags inserted after each span, e.g., <PLAUS> after “Yes, this is a plausible distractor”. Annotating an entire trace in a single pass yields poor results due to length, so we split each trace into chunks and annotate each chunk

⁴For example, we initially obtained separate codes for utterances such as “we need to solve correctly” and “but remember the correct answer is ...”. As both refer to the same underlying concept, namely explicitly referencing the correct answer (CORR), we merged them into a single category.

Strategy		Definition
<i>Understanding</i>		
INTER	Task Interpretation	Clarifying the problem statement / instructions to determine the required output.
CORR	Correct Answer Ref.	Retrieving, deducing, or justifying the correct answer to the MCQ.
<i>Candidate Construction</i>		
LINK	Conceptual Link	Establishing the relation distractors should share with the correct answer.
ERR_DESC	Error Description	Abstract description of a learner action or misconception (“a student might...”).
ERR_SIM	Error Simulation	Step-by-step simulation of a learner’s flawed reasoning.
INST	Outcome Instantiation	Any mention of a candidate distractor or incorrect answer.
<i>Quality Control and Selection</i>		
PLAUS	Plausibility Check	Assessing if or why a candidate would be tempting/believable to a student.
DISCR	Discriminability Check	Evaluating whether a candidate is clearly incorrect and unambiguous.
CURATE	Final Set Curation	Selecting, refining, or organizing candidates to improve overall set quality.
<i>Reconsideration (Cross-cutting)</i>		
RECON	Reconsideration	Challenging or revisiting a previous assumption (“but wait...”).

Table 1: A taxonomy for distractor generation grounded in learning-science literature and empirical observations of LLMs’ reasoning traces, which we use to annotate reasoning traces as shown in Fig. 1. Appendix Table 15 illustrates each strategy with trace excerpts.

separately; to preserve problem-specific context across chunks, we augment each chunk’s prompt with examples extracted from the full trace (see Appendix §A.5 for details). Experimental details and validation results are reported in §4.2.

4 Experiments

We first establish that LLM’s reasoning strategies help with distractor generation (§4.1), then examine how LLMs use and orchestrate the strategies of our taxonomy (§4.2), and finally discuss alignment with best practices and failure modes (§4.3).

Datasets. For both taxonomy construction and experiments, we use the Eedi Math (King et al., 2024) and SciQ (Welbl et al., 2017) datasets, both MCQ datasets with three distractor options per item. Eedi serves as our primary dataset, as it contains grade-school mathematics problems with annotated high-level student misconceptions and is widely used in prior work on distractor generation and misconception modeling (Feng et al., 2024; Lee et al., 2024; Parikh et al., 2025). We also include SciQ, a science QA dataset, to validate that our findings generalize across domains. While it does not come with misconception annotations, these are not required for taxonomy construction or our main trace-level analysis. We remove image-based questions and questions that depend on answer choices (e.g., “Which of the following is correct?”; see Appendix §A.1 for further details). For SciQ, we randomly sample 500 questions; for Eedi, we keep only MCQs with annotated misconceptions, yielding 429 problems.

Models and prompting. We use two reasoning models, DEEPSEEK-V3.2 (DeepSeek-AI, 2025) and GLM-4.7 (GLM-4.5 Team, 2025), across the two datasets and three prompting settings: (i) **direct prompting**, where reasoning is disabled and models directly output distractors; (ii) **chain-of-thought (CoT) prompting** (Wei et al., 2022), where models are prompted to think step-by-step before generating distractors but reasoning is disabled; (iii) **reasoning prompting**, where models generate additional reasoning tokens before outputting distractors. Only settings (ii) and (iii) produce intermediate traces that can be analyzed. All settings share the same base **neutral prompt** for distractor generation, with setting-specific adaptations described in Appendix §A.3; the LLM is prompted for each MCQ separately.

Performance metrics. To stratify traces by performance and evaluate the effect of reasoning on LLM distractor generation (§4.1), we define three evaluation metrics. Each dataset $\{p_n\}_{n=1}^N$ contains N problems $p_n = (x_n, y_n, \mathcal{Z}_n)$, where x_n is the textual description of the problem, y_n is the correct answer choice, and $\mathcal{Z}_n$ with $|\mathcal{Z}_n| = 3$ is the set of (three) ground-truth distractors. For the problem indexed by n , we sample candidate distractors from an LLM, yielding a multiset $\hat{\mathcal{Z}}_n = (\hat{\mathcal{Z}}_n, m_n)$, where $\hat{\mathcal{Z}}_n$ is the underlying set and $m_n: \hat{\mathcal{Z}}_n \rightarrow \mathbb{N}$ is the multiplicity of the elements in $\hat{\mathcal{Z}}_n$ (i.e., their counts); we write $|\hat{\mathcal{Z}}_n| \stackrel{\text{def}}{=} \sum_{z \in \hat{\mathcal{Z}}_n} m_n(z)$ for the total candidate count. In practice, we restrict $|\hat{\mathcal{Z}}_n|$ to 3 via prompting and post-processing. We use three metrics: **proportional match** $|\mathcal{Z}_n \cap \hat{\mathcal{Z}}_n|/|\mathcal{Z}_n|$ (Feng et al., 2024; Fernandez et al., 2024), the

Model	Setting	Eedi	SciQ
DS-V3.2	Direct	0.34 ± 0.03	0.08 ± 0.02
	CoT	0.51 ± 0.03	0.13 ± 0.02
	Reasoning	0.52 ± 0.03	0.14 ± 0.02
GLM-4.7	Direct	0.42 ± 0.03	0.12 ± 0.02
	CoT	0.50 ± 0.03	0.15 ± 0.02
	Reasoning	0.52 ± 0.03	0.15 ± 0.02

Table 2: Proportional match (mean $\pm$ 95% CI based on t-distribution) for DEEPSEEK-V3.2 (DS-V3.2) and GLM-4.7, on the Eedi and SciQ datasets, under different prompting settings.

number of candidates that equal the correct answer choice (**#correct**) $m_n(y_n)$ (max value 3, optimal value 0), and the number of redundant candidate choices (**#repetitions**) $|\hat{Z}_n| - |\hat{Z}_n|$ (max value 2, optimal value 0). We report the average across all N problem instances and confidence intervals (CIs) based on t-distributions. These metrics require equivalence checks between distractors, which are arbitrary token sequences and not restricted to numerical values. Following prior work (Ross and Andreas, 2025), we use an LLM as judge. gpt-4-1-mini (OpenAI Team, 2024) achieves a macro-average precision of 98% and recall of 96%, which we verified through manual validation of 1000 random comparisons. Further details are provided in Appendix §A.2.

4.1 Reasoning Improves Performance

Before examining reasoning traces, we first assess what output-level performance metrics reveal about the usefulness of reasoning for LLMs’ distractor generation. We find that proportional match improves significantly when moving from direct prompting to either CoT or reasoning for both LLMs and datasets (Table 2). Furthermore, as detailed in Appendix §B.1, on Eedi **#correct** decreases by $\approx 92\%$ and **#repetitions** by 82%; on SciQ **#correct** drops by 82% while **#repetitions** remain low overall. These patterns suggest that the additional tokens contain useful distractor-generation strategies, with the drops in **#correct** and **#repetitions** pointing to successful discrimination checks and curation. We interpret the substantially higher proportional match on Eedi than on SciQ as partly reflecting task structure: misconception-based construction may constrain the output space to specific plausible errors, making alignment with human-authored distractors more tractable.

4.2 The LLM Distractor Generation Process

We present three complementary views of LLMs’ distractor-generation processes: overall strategy usage, how strategy usage evolves throughout a trace, and how strategies are organized into reasoning subprocesses. We report DEEPSEEK-V3.2 with reasoning enabled; reported trends are consistent across GLM-4.7 and the CoT setting, with full results in Appendix §§ B.2 to B.4.

Annotation. We apply the pipeline from §3.3 to 885 traces, stratified by dataset, model, CoT versus reasoning, and performance (proportional match above vs. below 0.5). We validate annotations on a representative subset of 64 traces using GPT-4.1 as annotator, achieving a precision of 0.97 and recall of 0.95, macro-averaged across strategy tags, datasets, and models; a detailed breakdown is in Appendix §A.5.

Occurrences of strategies. We analyze the average number of annotated spans per trace for each strategy of our taxonomy in Table 3. INST is among the most frequently observed strategies across both datasets,⁵ and INTER, CORR, PLAUS, and CURATE also occur consistently. By contrast, LINK is prevalent on SciQ but nearly absent on Eedi, whereas ERR_DESC and ERR_SIM dominate on Eedi but appear only sporadically on SciQ. DISCR and RECON show smaller but consistent differences across datasets: DISCR is more frequent on SciQ and RECON on Eedi. The consistent presence of INTER, CORR, INST, PLAUS, and CURATE across both datasets suggests a shared high-level structure involving problem understanding, candidate construction, evaluation and selection. At the same time, the dataset-specific patterns are consistent with two distinct approaches to candidate construction and evaluation: misconception-based on Eedi, where ERR_DESC and ERR_SIM drive candidate generation, and similarity-based on SciQ, where candidates are identified via LINK and their incorrectness is verified via DISCR.

Strategies over time. We track the proportion of each strategy across five temporal bins, where temporal position within each trace is normalized to $[0, 1]$ by character offset. The resulting graphs in the top row of Fig. 2 reveal clear shifts in strategy usage throughout the reasoning process. Early

⁵This is expected, given that repeated occurrences are counted separately and statements involving PLAUS, DISCR and CURATE often reference INST.

Strategy	Eedi	SciQ
Task Interpretation	11.22 $\pm$ 1.38	5.84 $\pm$ 0.80
Correct Answer Ref.	9.33 $\pm$ 1.40	5.52 $\pm$ 0.89
Conceptual Link	0.04 $\pm$ 0.05	5.36 $\pm$ 0.87
Error Description	22.83 $\pm$ 2.70	0.78 $\pm$ 0.31
Error Simulation	5.70 $\pm$ 1.18	0.00 $\pm$ 0.00
Outcome Instantiation	34.84 $\pm$ 4.05	24.71 $\pm$ 2.95
Plausibility Check	7.94 $\pm$ 1.15	5.33 $\pm$ 0.90
Discriminability Check	2.54 $\pm$ 0.59	10.34 $\pm$ 1.50
Final Set Curation	3.34 $\pm$ 0.39	5.83 $\pm$ 0.60
Reconsideration	5.46 $\pm$ 0.71	1.05 $\pm$ 0.23

Table 3: Average occurrences of taxonomy strategies (Table 1) in reasoning traces for DEEPSEEK-V3.2 (mean $\pm$ 95% CI based on t-distribution) on both datasets.

reasoning is usually dominated by task interpretation (INTER) and identifying the correct answer (CORR), while usage of final set curation (CURATE) increases towards the end. For Eedi, error description (ERR_DESC) peaks right after INTER and CORR, also coinciding with an increase in instantiation of incorrect outcomes (INST). Toward the end, plausibility assessment (PLAUS) becomes more frequent. For SciQ, conceptual linking (LINK) appears more frequently in the beginning, while plausibility (PLAUS) and discriminability checks (DISCR) are performed across the middle and later stages. These temporal profiles are consistent with a progression from problem understanding to candidate construction to evaluation and selection. We observe similar trends for CoT but with INTER and CORR taking up a larger share of the reasoning process, consistent with fewer distractor candidates being explored (see Appendix §B.3).

Common sequences of strategies. To analyze the typical sequences in which strategies appear, we examine transition probabilities between strategies within sliding windows of 4 consecutive strategy annotations, shown in the bottom row of Fig. 2. The transitions reveal functional groupings. On Eedi, ERR_DESC and INST are tightly coupled. From the traces we examined during open coding and annotation validation, we observe two recurring sub-patterns: models either list multiple error descriptions before instantiating them, or iteratively interleave the two, sometimes with PLAUS. On SciQ, the coupled trio is INST, PLAUS, and DISCR, reflecting a quality-control focus in the construction phase. Self-transitions for INTER and CORR indicate that these problem-understanding strategies tend to extend across multiple consecutive annotation spans. For CoT on Eedi (see Appendix §B.4), a common

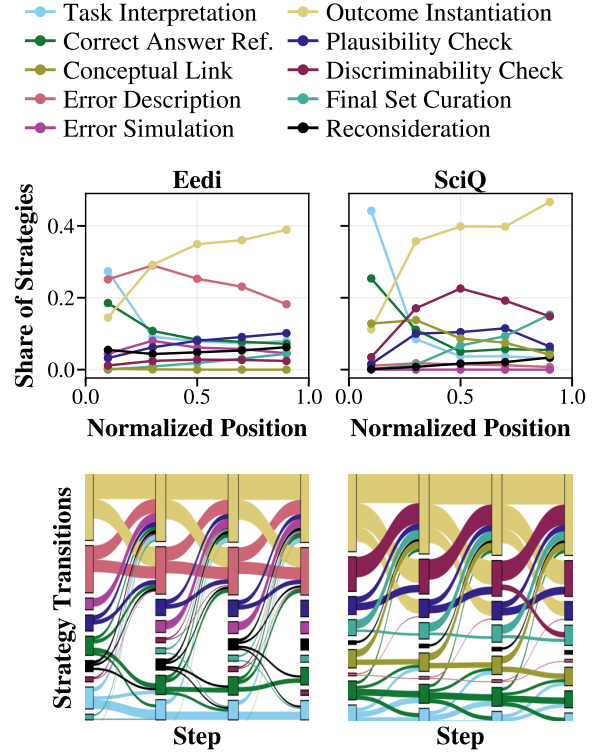


Figure 2: Strategy usage in DEEPSEEK-V3.2 reasoning traces on Eedi, shown in two views (strategies defined in Table 1). Top: temporal distribution across five normalized stages over the span of a reasoning trace (0 = start, 1 = end), where proportions sum to one within each bin. Bottom: transition structure between strategies over sequences of length 4 (left to right), where node height reflects strategy prevalence and edge width encodes transition probability. Only dominant transitions (>15% outgoing probability) are shown for clarity.

ERR_DESC $\rightarrow$ ERR_SIM $\rightarrow$ INST transition suggests that LLMs explicitly simulate the erroneous reasoning step, possibly ensuring consistency between the error description and its instantiation.

4.3 Discussion and Implications

Shared strategy structure and domain adaptation. Across models, datasets, and prompts, the analyses above converge on a shared high-level distractor-generation pipeline: problem understanding, candidate construction, evaluation and selection. This pipeline is exploratory: after anchoring to the correct answer, models over-generate candidate distractors and then select a final set, resembling a best-of- N search. DEEPSEEK-V3.2 in reasoning mode considers, on average, 12.6 alternative incorrect solution paths on Eedi and 9.9 on SciQ before final selection, compared to 5.9 and 5.5 under CoT prompting. These counts are based on manual analysis of 16 randomly selected

problems per setting. What differs across domains is the candidate construction and evaluation stage. We hypothesize this reflects task structure: on Eedi, where distractors are anchored to annotated student misconceptions, describing and instantiating these errors (ERR_DESC, ERR_SIM, INST) mostly yields already-incorrect candidates, reducing the need for explicit discriminability checks. On SciQ, models instead use LINK to construct candidates via semantic relatedness to the correct answer and DISCR to verify they are indeed incorrect, consistent with LINK appearing early in the trace and DISCR later.

Comparison with literature best practices. On Eedi, where misconception-based construction is both applicable and the learning-science-preferred approach, we can directly test whether LLMs actually follow this approach. The strategy distributions and temporal profiles indicate that they predominantly do, following a solve-first, error-injection process in which a potential student error is articulated, instantiated through erroneous reasoning, and used to produce a distractor, rather than relying on surface-level similarity heuristics. A targeted trace-level analysis further confirms this (see Appendix §C.1): models construct a step-by-step solution first in 95.2% of traces on average across models, and among these solve-first traces, diverge at a specific step to inject an error in 73.8% of cases. Conversely, similarity-based strategies are rare ($\approx 3\%$ of traces) and largely confined to narrow settings such as rounding or magnitude-estimation problems. Beyond this structural alignment, we also examine the cognitive stance models adopt when reasoning about errors: when prompted to simulate a specific student error, 99.3% of DEEPSEEK-V3.2 traces reason from an expert perspective (e.g., “the student will likely do...” or “I must determine what the student would likely do...”) rather than adopting a student’s first-person stance (see Appendix §C.2).

Diagnosing failure modes. We analyze multiple stages of LLMs’ distractor-generation pipeline to determine whether failures stem from (i) incorrect error simulation, (ii) the absence of an effective reasoning structure, (iii) failure to recover the correct solution, or (iv) inaccurate plausibility assessment and candidate selection. We find that the first two rarely fail, whereas the latter two are the dominant bottlenecks, as detailed below.

(i) Error simulation rarely fails. When prompted to simulate a problem-specific student

error (sampled from Eedi’s misconception annotations for each of the 429 problems), DEEPSEEK-V3.2 achieves a mean accuracy of 0.92 ± 0.02 , manually verified against *any* faithful interpretation of the error (procedure in Appendix §C.2). However, this does not guarantee that the model selects the most plausible interpretation: only 62% of distractors judged faithful to a given error type appear in the human-authored ground-truth set, suggesting multiple valid instantiations exist. Where the targeted distractor was not produced, the underlying problems tend to have longer correct solutions ($p < 0.05$, Mann-Whitney U test, comparing solution step counts), suggesting the model is less likely to reproduce the targeted distractor on problems with longer solutions.

(ii) Reasoning structure rarely fails. A literature-informed prompt enforcing explicit multi-step reasoning (solving, enumerating errors, simulating them, evaluating plausibility, and selecting distractors; prompt in Appendix §C.3) yields no significant improvement over the neutral prompt for either model (DEEPSEEK-V3.2: $0.52 \pm 0.03 \rightarrow 0.55 \pm 0.02$; GLM-4.7: $0.52 \pm 0.03 \rightarrow 0.52 \pm 0.03$; paired t -test, $p > 0.05$), suggesting that both models already self-organize an effective reasoning structure without explicit procedural guidance. This observation largely generalizes across seven further reasoning models (Appendix §C.3).⁶

(iii) Correct-solution recovery is a genuine bottleneck. Providing the correct answer to the MCQ in the prompt improves proportional match on both Eedi ($0.52 \pm 0.03 \rightarrow 0.56 \pm 0.03$) and SciQ ($0.14 \pm 0.02 \rightarrow 0.18 \pm 0.03$; see footnote footnote 3), confirming that unreliable solution recovery propagates through the pipeline and degrades downstream generation.

(iv) Candidate selection is a second bottleneck. We can quantify how much the model loses at selection by asking how many human-authored distractors it proposes somewhere in its reasoning but leaves out of its final three. Proportional match already counts how many reach the final set. To count how many the model proposes at any point instead,

⁶The exception is the OpenAI models, for which enforcing the structure improves performance: GPT-OSS-20B, GPT-OSS-120B (OpenAI Team, 2025), and GPT-5 (OpenAI Team, 2026) benefit, whereas GEMMA-3 (Google Gemma Team, 2025), GEMINI-2.5 (FLASH/PRO) (Google Gemini Team, 2025), and GLM-4.7 FLASH (GLM-4.5 Team, 2025) do not. As the OpenAI models are also the lowest-scoring under the neutral prompt, model family and performance headroom are confounded.

we introduce an *oracle* ranker that matches each annotated INST candidate against the human-authored set (see Appendix §C.4); the gap between the two is the selection loss. The oracle raises DEEPSEEK-V3.2’s proportional match from 0.52 to 0.67 on Eedi and from 0.14 to 0.29 on SciQ (paired t -test, $p < 10^{-7}$), recovering about a third of the otherwise-missed distractors on Eedi and a sixth on SciQ. This suggests the model frequently proposes a valid distractor and then discards it. Part of this loss might be attributable to the model’s own plausibility and discriminability checks (PLAUS, DISCR): resolving each check to the candidate it evaluates and classifying whether it argues for or against keeping it (details in Appendix §C.5), we find that the model argues against keeping 28% of the candidates that match human-authored ones on Eedi and 15% on SciQ under DEEPSEEK-V3.2 (the same pattern holds for GLM-4.7). The checks do discriminate, rejecting invalid candidates 1.5–1.9 $\times$ as often as a valid ones, but they still turn a non-trivial share of valid candidates negative. Even a perfect ranker cannot recover a distractor the model never proposed, and 33% of human-authored distractors on Eedi and 71% on SciQ never appear in the reasoning at all. These misses point to a failure of candidate *generation* rather than selection, especially on SciQ.

Characteristics of high- and low-scoring traces. The stage analyses above pool all traces. We can instead split individual traces by output quality into high- and low-scoring at a proportional-match threshold of 0.5. On Eedi the strategy profiles differ between the two groups but not on SciQ, and the difference on Eedi is concentrated in one strategy: low-scoring traces enumerate student errors (ERR_DESC) more. This is significant under reasoning and borderline under CoT.⁷ Low-scoring traces are longer and may come from harder problems, but the gap is neither a length nor a difficulty effect: it holds as a share of strategies and under grouped solution-step counts. Two strategies, correct-answer reference (CORR) and task interpretation (INTER), have a higher *share* in high-scoring traces but no higher count: these traces contain fewer tags overall, so a similar count forms a larger fraction, not more of either strategy (§C.6). The

error-enumeration gap itself may have a mechanical source: problems with a larger space of plausible errors may draw more enumeration and make any single human-authored distractor statistically harder to match.

5 Conclusion

We introduced a taxonomy of ten distractor-generation strategies grounded in learning-science principles and observed model behavior, and applied it to reasoning traces from two LLMs across math and science MCQs.

Across all settings, we find a shared high-level structure involving problem understanding, anchoring to the correct solution, exploratory candidate construction, and iterative evaluation and selection, resembling a best-of- N search over incorrect solution paths. On Eedi, where all human-annotated distractors reflect specific misconceptions, models follow a solve-first, error-injection approach, constructing a step-by-step solution in 95.2% of traces and diverging to inject an error in 73.8% of those cases. This is in close alignment with the preferred misconception-based approach in the educational literature. However, on SciQ, they instead rely more on semantic similarity and explicit discriminability checks.

Not all parts of this distractor-generation pipeline are equally reliable, however: error simulation and the implicit organization of multi-step reasoning rarely fail, while failures concentrate in correct-solution recovery and downstream candidate selection. Providing the correct solution in the prompt yields significant relative improvements of 6.4% on Eedi and 30.7% on SciQ (see footnote 3), confirming correct-solution recovery as a primary bottleneck.

Together, these findings offer a concrete answer to how LLMs arrive at incorrect-but-plausible reasoning: through a structured but domain-adaptive process that mirrors expert practice on math problems while exposing clear points of failure across both domains. By isolating these failure points, our analysis points to targeted interventions, with implications for automated assessment, intelligent tutoring systems, and scalable creation of high-quality educational content.

Limitations

Our analysis is subject to several limitations. First, our analysis characterizes *what the reasoning trace*

⁷Significance comes from permutation tests: PERMANOVA (Anderson, 2001; McArdle and Anderson, 2001) for whether the overall ten-strategy profile differs, then one test per strategy with the significance threshold tightened for running ten of them (see Appendix §C.6).

says, not necessarily *what the model internally computes*. Reasoning traces are not guaranteed to faithfully reflect the underlying decision process, so the observed strategies should be interpreted as descriptions of externalized reasoning rather than mechanistic claims. Prior work argues, however, that reasoning traces are structured and shape model outputs rather than being mere post-hoc narration (Marjanovic et al., 2026).

Second, both studied models are open-weight reasoning models. Reasoning models are the natural starting point for process-level analysis, as they produce explicit intermediate traces that can be directly annotated with strategies; however, non-reasoning LLMs and closed-source models may exhibit different strategy distributions, and our quantitative findings should not be extrapolated to those settings without further study.

Finally, performance is measured against a finite set of human-annotated distractors, which does not account for further plausible distractors that are absent (see §4.3). On SciQ, this concern is more acute: distractors were not curated to reflect specific student misconceptions, making proportional match a weaker benchmark signal on that dataset. Both issues primarily affect proportional match as a benchmark metric; our core contribution is the process-level analysis, which does not depend on ground truth completeness.

Acknowledgments

We thank Eedi and the creators of SciQ for making their datasets publicly available.

References

- Elaf Alhazmi, Quan Z. Sheng, Wei Emma Zhang, Munazza Zaib, and Ahoud Alhazmi. 2024. *Distractor generation in multiple-choice tasks: A survey of methods, datasets, and evaluation*. In *EMNLP*, pages 14437–14458.
- Marti J. Anderson. 2001. *A new method for non-parametric multivariate analysis of variance*. *Austral Ecology*, 26(1):32–46.
- Robert B. Ashlock. 2010. *Error patterns in computation: Using error patterns to help each student learn*. Allyn & Bacon.
- Luca Benedetto, Giovanni Aradelli, Antonia Donvito, Alberto Lucchetti, Andrea Cappelli, and Paula Buttery. 2024. *Using LLMs to simulate students’ responses to exam questions*. In *EMNLP*, pages 11351–11368.
- Yoav Benjamini and Yosef Hochberg. 1995. *Controlling the false discovery rate: A practical and powerful approach to multiple testing*. *Journal of the Royal Statistical Society: Series B (Methodological)*, 57(1):289–300.
- John Seely Brown and Kurt VanLehn. 1980. *Repair theory: A generative theory of bugs in procedural skills*. *Cognitive science*, 4(4):379–426.
- Darryl J. Chamberlain and Russell Jeter. 2020. *Creating diagnostic assessments: Automated distractor generation with integrity*. *The Journal of Assessment in Higher Education*.
- Matthew M. Chingos. 2012. *Strength in numbers: State spending on K-12 assessment systems*. Brown Center on Education Policy, The Brookings Institution.
- Juliet Corbin and Anselm Strauss. 2014. *Basics of qualitative research: Techniques and procedures for developing grounded theory*. Sage publications.
- Jimmy De La Torre. 2009. *A cognitive diagnosis model for cognitively based multiple-choice options*. *Applied Psychological Measurement*, 33(3):163–183.
- DeepSeek-AI. 2025. *DeepSeek-V3.2: Pushing the frontier of open large language models*. *arXiv preprint arXiv:2512.02556*.
- Jacob Doughty, Zipiao Wan, Anishka Bompelli, Jubahed Qayum, Taozhi Wang, Juran Zhang, Yujia Zheng, Aidan Doyle, Pragnya Sridhar, Arav Agarwal, Christopher Bogart, Eric Keylor, Can Kultur, Jaromir Savelka, and Majd Sakr. 2024. *A comparative study of AI-generated (GPT-4) and human-crafted MCQs in programming education*. In *Proceedings of the 26th Australasian Computing Education Conference*, pages 114–123.
- Soukaina Ezzaki, Najat Messaoudi, and Jaafar Khalid Naciri. 2025. *Evaluating educational content through virtual student simulations using large language models*. In *International Symposium on Generative AI and Education*, pages 241–250.
- Yael Feldman-Maggor, Mutlu Cukurova, Carmel Kent, and Giora Alexandron. 2025. *The impact of explainable AI on teachers’ trust and acceptance of AI EdTech recommendations: The power of domain-specific explanations*. *International Journal of AIED*, 35(5):2889–2922.
- Wanyong Feng, Jaewook Lee, Hunter McNichols, Alexander Scarlatos, Digory Smith, Simon Woodhead, Nancy Ornelas, and Andrew Lan. 2024. *Exploring automated distractor generation for math multiple-choice questions via large language models*. In *NAACL*, pages 3067–3082.
- Nigel Fernandez, Alexander Scarlatos, Wanyong Feng, Simon Woodhead, and Andrew Lan. 2024. *DiVERT: Distractor generation with variational errors represented as text for math multiple-choice questions*. In *EMNLP*.

- David Freedman and David Lane. 1983. [A nonstochastic interpretation of reported significance levels](#). *Journal of Business & Economic Statistics*, 1(4):292–298.
- Mark J. Gierl, Okan Bulut, Qi Guo, and Xinxin Zhang. 2017. [Developing, analyzing, and using distractors for multiple-choice tests in education: A comprehensive review](#). *Review of educational research*, 87(6):1082–1116.
- GLM-4.5 Team. 2025. [GLM-4.5: Agentic, reasoning, and coding \(ARC\) foundation models](#). *arXiv preprint arXiv:2508.06471*.
- Google Gemini Team. 2025. [Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities](#). *arXiv preprint arXiv:2507.06261*.
- Google Gemma Team. 2025. [Gemma 3 technical report](#). *arXiv preprint arXiv:2503.19786*.
- Thomas M. Haladyna, Steven M. Downing, and Michael C. Rodriguez. 2002. [A review of multiple-choice item-writing guidelines for classroom assessment](#). *Applied Measurement in Education*.
- Thomas M. Haladyna and Michael C. Rodriguez. 2013. [Developing and validating test items](#). Taylor & Francis.
- Ching Nam Hang, Chee Wei Tan, and Pei-Duo Yu. 2024. [MCQGen: A large language model-driven MCQ generator for personalized learning](#). *IEEE Access*, 12:102261–102273.
- Hyoungwook Jin, Minju Yoo, Jeongeon Park, Yokyung Lee, Xu Wang, and Juho Kim. 2025. [TeachTune: Reviewing pedagogical agents against diverse student profiles with simulated students](#). In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, pages 1–28.
- Sharon Judge, Jonna Bobzien, Ann Maydosz, Sabra Gear, and Petros Katsioloudis. 2013. [The use of visual-based simulated environments in teacher preparation](#). *Journal of Education and Training Studies*, 1(1).
- Tanja Käser and Giora Alexandron. 2024. [Simulated learners in educational technology: A systematic literature review and a Turing-like test](#). *International Journal of Artificial Intelligence in Education*, 34(2):545–585.
- Jules King, L Burleigh, Simon Woodhead, Panagiota Kon, Perpetual Baffour, Scott Crossley, Walter Reade, and Maggie Demkin. 2024. [Eedi - mining misconceptions in mathematics](#). Kaggle.
- Jaewook Lee, Digory Smith, Simon Woodhead, and Andrew Lan. 2024. [Math multiple choice question generation via human-large language model collaboration](#). In *Proceedings of the 17th International Conference on Educational Data Mining*, pages 941–946, Atlanta, Georgia, USA.
- Unggi Lee, Sanghyeok Lee, Junbo Koh, Yeil Jeong, Haewon Jung, Gyuri Byun, Yunseo Lee, Jewoong Moon, Jieun Lim, and Hyeoncheol Kim. 2023. [Generative agent for teacher training: Designing educational problem-solving simulations with large language model-based agents for pre-service teachers](#). In *NeurIPS’23 Workshop on Generative AI for Education (GAIED)*.
- Naiming Liu, Shashank Sonkar, and Richard Baraniuk. 2025. [Do LLMs make mistakes like students? Exploring natural alignments between language models and human error patterns](#). In *International Conference on AIED*, pages 364–377.
- Naiming Liu, Shashank Sonkar, Zichao Wang, Simon Woodhead, and Richard G. Baraniuk. 2023. [Novice learner and expert tutor: Evaluating math reasoning abilities of large language models with misconceptions](#). *arXiv preprint arXiv:2310.02439*.
- Ulrich Ludewig, Jakob Schwerter, and Nele McElvany. 2023. [The features of plausible but incorrect options: Distractor plausibility in synonym-based vocabulary tests](#). *Journal of Psychoeducational Assessment*, 41(7):711–731.
- Jakub Macina, Nico Daheim, Sankalan Chowdhury, Tanmay Sinha, Manu Kapur, Iryna Gurevych, and Mrinmaya Sachan. 2023. [MathDial: A dialogue tutoring dataset with rich pedagogical properties grounded in math reasoning problems](#). In *EMNLP*, pages 5602–5621.
- Sara Vera Marjanovic, Arkil Patel, Vaibhav Adlakha, Milad Aghajohari, Parishad BehnamGhader, Mehar Bhatia, Aditi Khandelwal, Austin Kraft, Benno Kroejer, Xing Han Lü, Nicholas Meade, Dongchan Shin, Amirhossein Kazemnejad, Gaurav Kamath, Marius Mosbach, Karolina Stanczak, and Siva Reddy. 2026. [DeepSeek-R1 thoughtology: Let’s think about LLM reasoning](#). *Transactions on Machine Learning Research*.
- Noboru Matsuda, William W. Cohen, Jonathan Sewall, Gustavo Lacerda, and Kenneth R. Koedinger. 2007. [Evaluating a simulated student using real students data for training and testing](#). In *User Modeling 2007*, pages 107–116.
- Brian H. McArdle and Marti J. Anderson. 2001. [Fitting multivariate models to community data: a comment on distance-based redundancy analysis](#). *Ecology*, 82(1):290–297.
- Robert J. Mislevy, Russell G. Almond, and Janice F. Lukas. 2003. [A brief introduction to evidence-centered design](#). *ETS Research Report Series*, 2003(1):i–29.
- Tanya Nazaretsky, Moriah Ariely, Mutlu Cukurova, and Giora Alexandron. 2022. [Teachers’ trust in AI-powered educational technology and a professional development program to improve it](#). *British Journal of Educational Technology*, 53(4):914–931.

- Andreas Opedal, Alessandro Stolfo, Haruki Shirakami, Ying Jiao, Ryan Cotterell, Bernhard Schölkopf, Abulhair Saparov, and Mrinmaya Sachan. 2024. [Do language models exhibit the same cognitive biases in problem solving as human learners?](#) In *Proceedings of the 41st International Conference on Machine Learning*, pages 38762–38778.
- OpenAI Team. 2024. [GPT-4 technical report](#). *arXiv preprint arXiv:2303.08774*.
- OpenAI Team. 2025. [gpt-oss-120b and gpt-oss-20b model card](#). *arXiv preprint arXiv:2508.10925*.
- OpenAI Team. 2026. [OpenAI GPT-5 system card](#). *arXiv preprint arXiv:2601.03267*.
- Nisarg Parikh, Alexander Scarlatos, Nigel Fernandez, Simon Woodhead, and Andrew Lan. 2025. [LookA-like: Consistent distractor generation in math MCQs](#). In *Proceedings of the 20th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2025)*, pages 294–311.
- Kennedy Quaigrain and Ato Kwamina Arhin. 2017. [Using reliability and item analysis to evaluate a teacher-developed test in educational measurement and evaluation](#). *Cogent Education*, 4(1):1301013.
- Donya Roeein, Sankalan Pal Chowdhury, Mariia Ereemeeva, Yuan Qin, Debora Nozza, Mrinmaya Sachan, and Dirk Hovy. 2026. [PATs: Personality-aware teaching strategies with large language model tutors](#). In *Findings of the Association for Computational Linguistics: EACL 2026*, pages 4186–4211.
- Alexis Ross and Jacob Andreas. 2025. [Learning to make MISTAKES: Modeling incorrect student thinking and key errors](#). *arXiv preprint arXiv:2510.11502*.
- Lawrence M Rudner. 2009. [Implementing the graduate management admission test computerized adaptive test](#). In *Elements of adaptive testing*, pages 151–165. Springer.
- Andreas Säuberli and Simon Clematide. 2024. [Automatic generation and evaluation of reading comprehension test items with large language models](#). In *Proceedings of the 3rd Workshop on Tools and Resources for People with READING Difficulties (READI) @ LREC-COLING 2024*, pages 22–37.
- Alexander Scarlatos, Wanyong Feng, Digory Smith, Simon Woodhead, and Andrew Lan. 2024. [Improving automated distractor generation for math multiple-choice questions with overgenerate-and-rank](#). In *Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2024)*, pages 222–231.
- Karen Scouller. 1998. [The influence of assessment method on students’ learning approaches: Multiple choice question examination versus assignment essay](#). *Higher education*, 35(4):453–472.
- Jinnie Shin, Qi Guo, and Mark J. Gierl. 2019. [Multiple-choice item distractor development using topic modeling approaches](#). *Frontiers in psychology*, 10:825.
- Marie Tarrant, James Ware, and Ahmed M. Mohammed. 2009. [An assessment of functioning and non-functioning distractors in multiple-choice questions: a descriptive analysis](#). *BMC medical education*, 9(1):40.
- Ana Paula Tomás and José Paulo Leal. 2013. [Automatic generation and delivery of multiple-choice math quizzes](#). In *Principles and Practice of Constraint Programming*, pages 848–863.
- Kurt VanLehn. 1990. *Mind bugs: The origins of procedural misconceptions*. MIT press.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. [Chain-of-thought prompting elicits reasoning in large language models](#). In *NeurIPS*.
- Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. [Crowdsourcing multiple choice science questions](#). In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 94–106.
- Tao Wu, Jingyuan Chen, Wang Lin, Mengze Li, Yumeng Zhu, Ang Li, Kun Kuang, and Fei Wu. 2025. [Embracing imperfection: Simulating students with diverse cognitive levels using LLM-based agents](#). In *Proceedings of the 63rd Annual Meeting of the ACL (Volume 1: Long Papers)*, pages 9887–9908.

A Supplementary Material - Experimental Setup

A.1 Preprocessing the Dataset

Removing problems relying on choices. Both the Eedi and SciQ datasets contain problems of the form “Which of the following is correct?”, which rely on the provided answer choices to be solvable. This is problematic for distractor generation because it implies an effectively infinite answer space. Therefore, we manually reviewed all problems in the dataset to determine whether they admit a unique solution without the provided choices, and removed those that do not.

A.2 Measuring Performance

Equivalence checks. The metrics defined in §4 require checking whether candidate distractors $\hat{\mathcal{Z}}$ are equivalent to the annotated distractors $\mathcal{Z}$. Requiring exact matches, while eliminating false positives, would be far too strict: most problems’ answers may be expressed in multiple ways (e.g., $\frac{1}{2}$ vs. $1/2$) or equivalence may depend on the problem description (e.g., $\frac{1}{2}$ is not equivalent to 0.5 if the problem asks for it to be converted to decimal). Following prior work, we use an LLM (GPT-4.1-MINI (OpenAI Team, 2024), prompted to judge equivalence of two provided answers) to judge semantic equivalence when exact matches come back negative (Ross and Andreas, 2025). Our policy is to enforce output-formatting constraints that a problem states explicitly, but not to infer constraints that are left unstated. For instance, if a problem explicitly requires its answer as a fully reduced expression, $\frac{6}{3}$ and 2 are treated as non-equivalent, whereas if it imposes no such requirement they are considered equivalent. This mirrors how the ground-truth distractors themselves are written and avoids penalizing candidates for surface form when the problem does not constrain it. Prompts are split into versions with examples specific to Eedi (Table 4) and to SciQ (Table 5). The LLM achieves 98% precision and 92% recall in equivalence judgments over 500 Eedi comparisons and 97% precision and 100% recall over 500 SciQ comparisons.

System Prompt

You are an AI assistant tasked with judging whether two answer choices to a middle-school multiple-choice math problem are semantically the same as one another. You must not solve the problem and not evaluate factual correctness – only compare the two answers with one another relative to the problem’s formatting requirements. Your output must follow this exact structure:
<format> [TRUE/FALSE] </format>

```
<equivalent> [TRUE/FALSE] </equivalent>
Meaning of <format>:
Output TRUE if the problem explicitly requires a specific
numeric format, such as:
- rounding to a given number of decimal places or
significant digits
- expressing the answer in scientific notation
- expressing the answer as a simplified fraction
- expressing the answer in terms of a constant (e.g., “in
terms of  $\pi$ ”)
- any other explicitly stated formatting requirement
Ignore unit requirements (e.g., “in cm” does NOT count as a
specific format).
Output FALSE if the problem does not explicitly specify a
numeric format.
Meaning of <equivalent>:
Determine whether answer_1 and answer_2 represent the same
value or concept under the rules:
If <format> is FALSE (no required numeric format):
Two answers are equivalent if:
- they have the same mathematical value (e.g.,  $3.1 = 31/10$ )
- they differ only in non-semantic aspects (e.g., LaTeX
wrappers, capitalization, filler words)
Examples of equivalent under <format> = FALSE:
- 10, 10.0,  $\backslash(10\backslash)$ 
- 3.1, 31/10
- fourteen, 14
- Only Bob, Bob
If <format> is TRUE (specific format required):
Two answers are equivalent only if:
1. They represent the same mathematical value, AND
2. They are both expressed in the required format.
This means:
- 3.14 vs. 3.140 (when rounding to 2 decimal places
required)  $\rightarrow$  not equivalent
-  $4\pi$  vs. 12.56 (when “in terms of  $\pi$ ” required)  $\rightarrow$  not
equivalent
- 3.1 vs. 31/10 (when “round to one decimal place” required)
 $\rightarrow$  not equivalent
Ignore trivial formatting wrappers (e.g.,
 $31/10 = \backslash(31/10\backslash)$ ).
General Rules:
- Do not solve the problem.
- Do not judge correctness of the answers.
- Only compare answer_1 with answer_2.
- answer_1 and answer_2 can be equivalent regardless of
whether they are correct or not.
```

User Prompt

```
<problem> {problem_formulation} </problem>
<answer_1> {answer_a} </answer_1>
<answer_2> {answer_b} </answer_2>
```

Table 4: Prompt for judging equivalence between the two answers $\{answer_a\}$ and $\{answer_b\}$ to a math multiple-choice question $\{problem_formulation\}$ of Eedi.

System Prompt

You are an AI assistant tasked with judging whether two answer choices to a middle-school multiple-choice science question are semantically the same as one another. You must not answer the question and not evaluate factual correctness – only compare the two answers with one another relative to what the question is asking. Your output must follow this exact structure:
<equivalent> [TRUE/FALSE] </equivalent>
Meaning of <equivalent>:
Two answers are equivalent if they refer to exactly the same scientific concept, entity, term, or value, differing only in:
- capitalization, punctuation, pluralization, articles, or filler words
- trivial formatting wrappers (e.g., LaTeX, extra whitespace)
- abbreviation vs. its expansion (e.g., DNA vs. deoxyribonucleic acid)
- well-established synonyms denoting the exact same referent (e.g., heat vs. thermal energy)
- rewording that preserves the exact same referent without adding or removing entities

Examples of equivalent: - photosynthesis = Photosynthesis - planet = a planet - mirrors = a mirror - DNA = deoxyribonucleic acid - heat = thermal energy - products = the products Two answers are NOT equivalent if: - they name different scientific concepts, even if closely related (e.g., photosynthesis vs. respiration, mirrors vs. lenses, circadian rhythm vs. circannual rhythm) - one is a subset or superset of the other, or introduces or removes entities (e.g., “protons and neutrons” vs. “protons”) - they give different specific values or quantities (e.g., “less than half” vs. “5%”, “type 1” vs. “type 2 diabetes”) - they specify different levels of generality that change the referent (e.g., “bacteria” vs. “E. coli”) Examples of NOT equivalent: - mirrors $\neq$ lenses - protons and neutrons $\neq$ protons - carbon $\neq$ nitrogen - heat $\neq$ ultraviolet General Rules: - Do not answer the question. - Do not judge correctness of the answers. - Only compare answer_1 with answer_2. - answer_1 and answer_2 can be equivalent regardless of whether they are correct or not.
User Prompt <pre><problem> {problem_formulation} </problem> <answer_1> {answer_a} </answer_1> <answer_2> {answer_b} </answer_2></pre>

Table 5: Prompt for judging equivalence between two answer choices $\{answer_a\}$ and $\{answer_b\}$ to a middle-school multiple-choice science question $\{problem_formulation\}$ of SciQ.

A.3 Collecting Reasoning Traces and Evaluating Answer Equivalence

We provide the prompt used for distractor generation in both the direct and reasoning settings (Table 6), as well as the chain-of-thought prompt that enables the non-reasoning models to generate distractors step-by-step (Table 7). For all models, we set token limits to 16k for reasoning and 8k for CoT.

System Prompt You will be given a $\{math, science\}$ question along with the correct answer. Please generate 3 incorrect distractor answers for the question to be used as multiple-choice options in a multiple-choice exam. [Template] Distractor1: ... Distractor3: User Prompt Question: $\{problem_formulation\}$ Answer: $\{correct_answer\}$
--

Table 6: Neutral prompt for generating distractors for one math problem $\{problem_formulation\}$. Parts in blue are shown only when the correct answer $\{correct_answer\}$ is revealed. We use **math** for Eedi and **science** for SciQ.

System Prompt You will be given a $\{math, science\}$ question. Please generate 3 incorrect distractor answers for the question to be used as multiple-choice options in a multiple-choice exam. Think step-by-step before giving your final answer. Output only your step-by-step reasoning and the final distractors like so: [Step-By-Step] Let's think step-by-step, ... [Final Answer] Distractor1: ... Distractor3: User Prompt Question: $\{problem_formulation\}$

Table 7: Neutral chain-of-thought (CoT) prompt for generating distractors for one math problem $\{problem_formulation\}$. We use **math** for Eedi and **science** for SciQ.

A.4 Details on Taxonomy Construction

During taxonomy construction, several additional and potentially complementary strategies were identified through both human and LLM-based open coding but were ultimately excluded. Table 8 summarizes these candidates, gives illustrative examples, their origin, and the rationale for exclusion.

Merging strategies and coverage. Although the final taxonomy does not mirror all strategies that surfaced during open coding exactly, conceptual correspondence between strategies was always clear when merging them. For example, the LLM proposed the strategy “Anchor on Correct Solution” which we merged into CORR. In cases where titles differed, example citations from the traces frequently helped clarify the mapping. For instance, the proposed strategy “Contextualize to Multiple-Choice Format” with citation from the reasoning trace:

“Usually, the options are formatted similarly to the correct answer. Since correct is 0.256, options like 2.56, 0.0256, and 1.12 are all in decimal form with a few digits.”

is naturally covered by CORR and CURATE.

Granularity of strategies. A key design decision in constructing the taxonomy concerns the level of granularity at which strategies are defined. For instance, PLAUS could be subdivided into more specific assessments, such as evaluating how common an error type is or identifying which constraints a distractor’s derivation violates. Although such

Strategy	Example / Description	Identified By	Reason for Exclusion
Factual Recall	Explicit recall of relevant background information	Human	Only observed as part of CORR.
Diagnostic Value	Stating the diagnostic value of a distractor (e.g., “tests the student’s attention to concept x”)	Human	Implicitly captured via ERR_DESC and LINK.
Properties of Good Distractors	Listing properties of good distractors.	Human	Only observed as part of INTER, LINK and CURATE.
Response Formatting	“We need to produce them in the format Distractor1:...”	Both	Reasoning about output format required by our specific prompt rather than a general reasoning strategy.
Recall Task-Specific Knowledge	“First, let’s recall the basic Roman numeral symbols ...”	Both	Explicitly observed only once and typically occurred implicitly within CORR or ERR_DESC.
Leveraging Domain-Specific Constraints	Enumerating limited answer space (e.g., “Who out of the two is correct?”).	Both	Problem-specific enumeration strategy; excluded to maintain focus on general reasoning strategies.
Exploration of Unusual Distractors	Claim that highly unusual or creative distractors were intentionally generated (no citation provided).	LLM	Conceptually overlaps with ERR_DESC and PLAUS; insufficiently well-defined as a standalone strategy.
Distractor Source Diversification	Explicit effort to ensure distractors differ in type or origin to avoid similarity.	LLM	Strong overlap with CURATE.
Verification and Self-Correction	Revisiting earlier step-by-step reasoning towards distractors, potentially correcting issues.	LLM	Overlaps with RECON and often difficult to distinguish from ERR_SIM.
Detailed Error Description	Captures specific behaviors such as incorrectly converting fractions to decimals without broader strategic framing.	LLM	Overlaps with ERR_SIM.

Table 8: Discarded candidate strategies during the open coding phase of the taxonomy construction. Note that both the manual open coding as well as the LLM surfaced many additional strategies that were strictly captured by existing strategies; we only list the ones that might capture complementary aspects compared to the strategies they were merged into.

finer-grained distinctions may offer additional analytical resolution, our objective is to capture *general* strategy patterns rather than highly specialized variants. Increasing granularity would also likely reduce annotation reliability, as additional categories introduce greater complexity and less sharply defined decision boundaries.

Automatic open coding. We show the full prompt used to let DEEPSEEK-V3.2 do open coding in Table 9.

Overview of literature sources of final taxonomy. The identified strategies largely align with prior literature. LINK, PLAUS, DISCR, and CURATE reflect item-writing guidelines that ensure distractors are semantically related to the correct answer, individually plausible, clearly incorrect, and collectively well-formed (Haladyna et al., 2002; Haladyna and Rodriguez, 2013; Gierl et al., 2017; Chamberlain and Jeter, 2020; Ludewig et al., 2023). CORR, ERR_DESC, ERR_SIM, and INST correspond to established distractor generation mechanisms, primarily misconception-based approaches (De La Torre, 2009; Ashlock, 2010; Shin et al., 2019; Tomás and Leal, 2013; Gierl et al., 2017), and are further supported by repair theory and accounts of systematic student errors (Brown and VanLehn, 1980; VanLehn, 1990). Finally, INTER and RECON capture general reasoning behaviors rather than distractor-specific mechanisms: INTER reflects interpretation of task requirements and output constraints under structured assessment design (e.g., Evidence-Centered Design) (Mislevy et al., 2003), while RECON reflects iterative backtracking and revision during solution generation.

System Prompt

You are an expert in cognitive task analysis, think-aloud protocol analysis, and grounded theory coding. Your task is to derive inductive categories of cognitive processes from reasoning traces of experts that are generating {math,science} distractors for a multiple choice exam. Follow qualitative analysis best practices: bottom-up coding, constant comparison, memoing, and grounding all claims in the data.

User Prompt

I will provide you with a list of reasoning traces. Your task is to discover common categories of reasoning or cognitive behaviors of the experts that are generating {math,science} distractors, using a systematic inductive approach. Your responsibilities:

1. Identify recurring cognitive behaviors or steps. 2. For every category, provide: - A clear definition (1-2 sentences) - A description of what behaviors fall under it 3. Provide 2-3 grounded citations for each category: - Verbatim excerpts from the traces - Include trace ID and step number (or text location) 4. Do not invent anything not

```
present in the traces. 5. Focus only on recurring
categories, not one-off behaviors.
OUTPUT FORMAT:
## Discovered Categories
### Category 1: [Name] **Definition:** [...]
**Example Citations:** - "... (Trace X, Step Y) - ..."
(Trace A, Step B) - "... (Trace C, Step D)
### Category 2: [Name] **Definition:** [...]
(Continue as needed.)
After listing categories, include:
## Notes on Method & Coverage Explain how the categories
were derived and how representative they are.
-
Here are the reasoning traces:
{traces_concatenated}
```

Table 9: Prompt used for open coding of expert reasoning traces. `{traces_concatenated}` is replaced with 10 double-new-line concatenated reasoning traces of either DEEPSEEK-V3.2 or GLM-4.7 generating distractors when prompted with the neutral prompt, as described in §3.2. We use **math** for Eedi and **science** for SciQ.

A.5 Annotation of Traces

As outlined in §3.3, we annotate all reasoning traces separately. Each trace represents an LLM’s reasoning process as it generated three distractors for a single math or science problem.

Automatic annotation. We use GPT-4.1 (OpenAI Team, 2024) with greedy decoding and maximum output length of 16k tokens as the annotation model. Our annotation pipeline (Alg. 1) proceeds in four steps: we first extract trace-specific grounded examples for each tag from the full trace, then split the trace into chunks, annotate each chunk (in batches) using the taxonomy and grounded examples, and finally concatenate the annotated chunks.

A single-pass annotation over the entire trace yields poor results due to length. Chunking addresses this, but individual chunks may lack problem-specific context established earlier in the trace, e.g., “This gets us $2 + 9m$ ” could be either INST or CORR without knowing the correct answer. To compensate, we extract grounded examples from the full trace and include them in the annotation prompt for every chunk, promoting consistent decision boundaries across chunks.

Example extraction. We prompt the LLM with the taxonomy and the complete trace (Table 10), instructing it to extract up to three representative spans per tag. Tags that do not appear in the trace are omitted from the example set.

Chunking. We split the trace on double line breaks once a segment exceeds 500 characters, applying a hard split if no double line break occurs within 2000 characters. Double line breaks were chosen because models typically use them to sep-

Algorithm 1 Annotation pipeline applied to a single reasoning trace, i.e., one LLM generating three distractors for one MCQ.

Require: trace t (LLM reasoning for one MCQ’s three distractors), taxonomy $\mathcal{T}$

```

1:  $E \leftarrow \text{EXTRACTEXAMPLES}(t, \mathcal{T})$  ▷ up to 3 spans per tag
2:  $C \leftarrow \text{CHUNKTEXT}(t)$  ▷ split on double line breaks
3:  $A \leftarrow []$ 
4: for batch  $B$  in  $C$  do
5:    $A.\text{EXTEND}(\text{ANNOTATECHUNKS}(B, E, \mathcal{T}))$ 
6: end for
7: return  $\text{CONCAT}(A)$ 

```

arate semantic units; the 500/2000 thresholds balance segment length against annotation quality.

Per-chunk annotation. We annotate each chunk individually, conditioned on the grounded examples and a dataset-specific description of our taxonomy for Eedi (Table 12) and SciQ (Table 13), used as part of the annotation prompt (Table 11). While the grounding is adapted to dataset-specific examples, the underlying decision boundaries remain consistent across descriptions.

Stratified sampling. We annotate 885 traces in total. For SciQ, the higher-performance stratification bucket comprises all 165 available traces and the lower-performance bucket comprises a random subset of 240 traces. For Eedi, both buckets contain 240 traces each.

System Prompt
You are a helper that extracts up to 3 short example spans for each taxonomy label from a single reasoning trace. For each tag, return a short examples block (plain text) with one line per tag in the following form: <TAG>: example1; example2 If no examples exist for a tag, use: <TAG>: (none) Return only the plain text block (no JSON, no commentary). TAXONOMY: {taxonomy_description}
User Prompt
TRACE START {trace} TRACE END Return only the examples block as described.

Table 10: Prompt for extracting short example spans for each strategy of the taxonomy, if present, from a reasoning trace $\{trace\}$. $\{taxonomy_description\}$ is replaced with the taxonomy description in Table 12 for Eedi and Table 13 for SciQ, respectively.

System Prompt
You are annotating a chunk of a thinking-out-loud protocol produced by an expert model with markers of a taxonomy. Context: The text is a verbalized reasoning trace of an expert generating incorrect distractor answers for a {math,science} multiple-choice question.

The expert’s task was: “You will be given a {math,science} question. Please generate 3 incorrect distractor answers for the question to be used as multiple-choice options in a multiple-choice exam.” The protocol contains the expert’s internal reasoning, planning, and candidate generation steps. Your job is to annotate this text by inserting taxonomy tags. Each marker marks the END of the smallest possible span that instantiates the category. TAXONOMY {taxonomy_description} EXAMPLES: {extracted_examples}
User Prompt
CHUNK START {trace_chunk} CHUNK END Return only the annotated chunk (no explanations).

Table 11: Prompt for annotating chunks of an LLM’s reasoning when generating distractors with strategies from our taxonomy. $\{taxonomy_description\}$ is replaced with the taxonomy description from Table 12 for Eedi and Table 13 for SciQ. $\{extracted_examples\}$ is replaced with the examples extracted using the prompt in Table 10. $\{trace_chunk\}$ is replaced with a chunk of the LLM’s distractor generation reasoning trace. We use **math** for Eedi and **science** for SciQ.

Taxonomy Description (Eedi)
<INTER> Definition: Reasoning about the task instructions or requirements – what the question asks for and what counts as valid answers. Rules: – Only mark when the expert revisits the task description and subsequently tries to gain clarity about the task itself. – Do NOT mark execution steps, calls to produce output, or listing candidates (e.g., “I’ll produce:”, “Let’s do:”, “Distractor1: 0.4<INST>”). Examples: – “We are given the question: ...” – “However<RECON>, the task is to generate three incorrect distractors, not the correct answer<INTER>” <LINK> Definition: Naming a conceptual axis or category from which candidate distractors are drawn. Rules: – Mark only the axis-preamble in an “axis: item1, item2, ...” enumeration. Each named item still gets <INST>. – Do NOT mark output-format/rendering meta (untagged): “options should be in decimal form”, “options formatted similarly to the correct answer”, “I should output exactly:”, “use the template”. – Do NOT mark error-category headers like “Common mistakes:” or “Common errors students make:” – these introduce ERR_DESC content. – Rare in math traces; most distractors come from error simulation (ERR_DESC/ERR_SIM), not conceptual-category enumeration. <CORR>

Definition: Correct computation or reasoning toward the correct solution for the question.

Rules: - Mark whenever correct reasoning or the correct answer is referenced. - If correct reasoning and errors are discussed together, mark both.

Examples: - " $2 \div 1/5 = 10$ <CORR>" - "Multiplying both sides by 4 gives $20 = k$ <CORR>, but a student might only multiply the numerator<ERR_DESC>"

<ERR_DESC>

Definition: High-level verbal description of a common mistake or misconception.

Rules: - Mark every description of an error.

Examples: - "A common mistake is forgetting to flip the fraction<ERR_DESC>" - " 46 <INST>(forgetting to add 2)<ERR_DESC>" - " $(x,y)=(-2,15)$ <INST>[from sign error<ERR_DESC>]" - "Mis-handling the negative<ERR_DESC>: $-10 + 8$ <ERR_SIM>= 2 <INST>"

<ERR_SIM>

Definition: Explicitly simulating incorrect reasoning.

Rules: - Mark when the expert simulates an incorrect calculation. - Single incorrect equations can be marked if they represent erroneous reasoning. - Mark the final incorrect outcome with <INST>. - ERR_DESC = a high level error description; ERR_SIM = a specific execution of an error

Examples: - " $5 - 2 = 3$, then add $1 =$ <ERR_SIM> 4 <INST>" - " $9 + 3 = 12$, write down 2, forget to carry the 1... final result <ERR_SIM> 82 <INST>" - "Convert the fraction incorrectly <ERR_DESC>: compute: $1 \frac{2}{3}$ <ERR_SIM><INST>"

<INST>

Definition: Any incorrect outcome (number, symbol, expression).

Rules: - Mark every candidate, even if later rejected. - Mark candidate values even when they appear inside task interpretation or reconsideration spans, as long as they name concrete answer options. - Each value in an enumeration of candidates is marked separately; enumeration markers like 1., 2., 3. are NOT tagged.

Examples: - " 0.4 <INST>, 0.1 <INST>, 2.5 <INST>" - "Possible answers could be Alice <INST>, Bob <INST>, etc" - " 980 <INST>might work" - " $(x,y)=(-2,15)$ <INST>[from sign error<ERR_DESC>]"

<PLAUS>

Definition: Judgment of how likely a student would choose an error or candidate.

Rules: - Mark plausibility comparisons or checks for incorrectness. - If also about final set, mark both PLAUS and CURATE.

Examples: - " 0.4 <INST>is more plausible than 0.1 <INST><PLAUS>" - "The student forgets to add?<ERR_DESC>Plausibly<PLAUS>" - " 0.4 <INST>is not a good distractor<PLAUS>" - "But is a student going to make that mistake?<PLAUS>"

<DISCR>

Definition: Verifying a candidate is incorrect / distinct from the correct answer (or rejecting one that's too close).

Rules: - Mark the verification act, not the candidate itself. - Triggers: distinctness checks ("all different", "X is correct, so we need incorrect"), per-candidate "but it's not the answer" tail-clauses (mid-sentence split), "too close to correct" rejections. - Distinct from PLAUS (would a student pick it?) and CURATE (final-set quality). - Do NOT fire on general "doesn't make sense" / "not sure" dismissal or on shape-based rejection ("not an integer"). DISCR asserts incorrectness or insufficient distinctness from the correct answer.

Examples: - " 0.256 is correct, so we need incorrect ones.<DISCR>" - " $8/3 \approx 2.66$ is too close to 2.828 and 2 .<DISCR>"

<CURATE>

Definition: Evaluation or selection of the final set of distractors (coverage, diversity, redundancy).

Rules: - Only mark when reasoning explicitly concerns the final set. - Otherwise, mark PLAUS.

Examples: - "Keep 0.4 <INST>and 2.5 <INST>, drop 0.1 <INST>to cover error types<CURATE>" - " 0.4 <INST>seems plausible<PLAUS>, keep that<CURATE>"

<RECON>

Definition: Reconsideration of a previous interpretation, candidate, plausibility judgment, or curation decision.

Rules: - Place <RECON>immediately after the cue word indicating reconsideration. - Marks the act of reconsidering, not the outcome. - Common cues: "actually", "alternatively", "instead", "however", "but wait", "on second thought", "reconsider"

Examples: - "Actually<RECON>, ..." - "Alternatively<RECON>, 980 <INST>could work<PLAUS>" - "On second thought<RECON>, that distractor is not likely<PLAUS>"

Table 12: The description of our taxonomy used for Eedi (Table 1) including rules and examples. We use this description as part of both the annotation (Table 11) and example extraction prompts (Table 10).

Taxonomy Description (SciQ)

Cross-cutting rules (abbreviated)

CC1. Tag every occurrence (no first-mention-only). CC2. Stack tags only when each genuinely applies. CC3. Output-formatting / template / phrasing talk is untagged [...]. CC4. Same fact gets different tags by function (e.g. <CORR> when grounding the answer vs. <LINK> when defining the design space). CC5. Place tags at clause boundaries, not at sentence ends. CC6. Pivot cues at sentence start (Instead, Alternatively, Actually, Wait, Another idea:, ...) ALWAYS trigger <RECON> immediately after the cue.

Tags

<INTER> - Engaging with what the task/question asks (interpreting, restating, breaking down). Switch to <CORR> once solving begins; section headers and output-formatting are not INTER.

Ex: "The question asks for the primary product of photosynthesis.<INTER>"

<CORR> - States, deduces, or grounds the correct answer. Place on the specific term span when named inline. Hedging meta toward committing to the answer is CORR.

Ex: "Glucose is the primary product of photosynthesis.<CORR>"

<LINK> - Conceptual/categorical design space for distractors, with NO specific candidate named. Specificity test: replace the noun with "such things"; if still a category-level criterion, it is LINK. Brainstorm-intro phrases ("Possible distractors could include:") are untagged, not LINK.

Ex: "Distractors should be other types of bonds.<LINK>"

<INST> - A specific value/term/concept named as a concrete answer-shaped option (drafted, brainstormed, or in a taxonomic enumeration). Each item in an enumeration is tagged separately. Axis-then-items bullets: axis gets <LINK>, each item gets <INST>. A correct-answer term in a candidate list still gets <INST> (plus <CORR>). [...]

Ex: "- By causative agent<LINK>: bacterial<INST>, viral<INST>, parasitic<INST>, fungal<INST>."

<PLAUS> - Why a specific candidate would be tempting (similarity, familiarity, commonness). Requires a named candidate (general "distractors should be plausible" is LINK). Common-misconception spans stack <ERR_DESC><PLAUS>. Dismissive scoping ("but not in the leg") is untagged.

Ex: "mitosis<INST> (a common confusion<PLAUS>)"

<DISCR> - Why a specific candidate is incorrect or contrasts with the correct answer. General "distractors must be wrong" is LINK; whole-set sanity checks are CURATE. Same dismissive-scoping caveat as PLAUS.

Ex: "It sounds plausible because of x<PLAUS> but is wrong due to y.<DISCR>"

<ERR_DESC> - High-level description of a cognitive error/misconception. The span MUST contain an action verb (confuse, mix up, forget, misapply, associate, ...); pure factual statements don't qualify.

Ex: "A student might associate mitosis with cell growth and forget it produces two identical daughter cells.<ERR_DESC>"

<ERR_SIM> - Step-by-step simulation of a misconception (vs. ERR_DESC's high-level description).

Ex: "A student might think: photosynthesis needs light, mitochondria use light too<ERR_SIM>, so the answer is mitochondria<INST>."

<CURATE> - Selecting/refining/finalizing the distractor set, including informal commits, end-of-trace selection, set-level sanity checks, and the closing line of a bare final-list block. Brainstorm-introduction phrases are NOT CURATE.

Ex: "Let's pick the three strongest ones.<CURATE>"

<RECON> - Reconsidering a prior interpretation/candidate/judgment/decision. Placed immediately after the pivot cue (never trailing). New content after <RECON> is tagged on its own.

Ex: "Instead<RECON>, I'll try a different category.<LINK>"

Decision tree (top-down; first match wins; co-tag only when each genuinely applies)

1. Reconsideration cue at sentence start? → <RECON>, then continue. 2. Names a specific candidate? → <INST>, then continue. 3. Why a candidate is tempting? → <PLAUS>. 4. Why a candidate is incorrect? → <DISCR>. 5. Cognitive error with action verb? → <ERR_DESC> (procedural → <ERR_SIM>). 6. Selecting/refining the final set? → <CURATE>. 7. Design-space, no candidate named? → <LINK>. 8. Engaging with the task? → <INTER>. 9. Grounding the correct answer? → <CORR>. Otherwise → untagged (output-formatting always falls here, per CC3).

Table 13: The SciQ taxonomy prompt, abbreviated. Long enumerations of trigger phrases, sub-patterns, and operational examples are elided as [...]; the full prompt is released with the code. We use this description as part of both the annotation (Table 11) and example extraction prompts (Table 10).

Annotation quality. We report detailed annotation performance—comparing agreement between human and GPT-4.1’s annotations—per strategy, on distractor-generation-traces of DEEPSEEK-V3.2 and GLM-4.7, across both datasets, and CoT versus reasoning mode in Table 14. Manual analyses were conducted on 64 traces, uniformly stratified across dataset, model, prompt, and high- versus low-performance cases. As a result, strategies that the models rarely engage in have very low support. However, the resulting uncertainty does not meaningfully affect the analyses in §4.2, since such strategies are far from playing a dominant role in the frequency analysis or dominant paths in the subprocess analysis. Overall agreement was high across most strategies.

A.6 Qualitative Examples

We give two complementary sets of examples, both drawn from DEEPSEEK-V3.2 and GLM-4.7 reasoning traces. Table 15 illustrates each taxonomy strategy (from Table 1) with short verbatim excerpts from the collected traces, to make the annotation concrete. Table 16 instead focuses on the generated *distractors*, classified by whether our equivalence judge (the same one underlying proportional match, §A.2) deems them equivalent to a human-authored distractor. It contrasts these high-quality distractors with low-quality ones that match no human-authored distractor (e.g., a candidate equal to the correct answer, or one not grounded in a coherent misconception), showing the full ground-truth and model distractor sets alongside the trace excerpt that produced or justified the focal candidate. Excerpts are shortened for space, with deletions marked [...]. Examples are from Eedi, except the LINK row in Table 15, which is near-absent on Eedi (Table 3) and is therefore drawn from SciQ.

B Full Results

Here, we present the full results for GLM-4.7 and for the CoT prompting setting in the experiments from §4.1 and §4.2. We also briefly compare these results to those of DEEPSEEK-V3.2 in reasoning mode.

B.1 Performance

In Table 17 we observe that both reasoning and CoT tend to reduce #correct and #repetitions over direct generation. On SciQ, #repetitions remain consistently low across settings.

B.2 Occurrences of Strategies

Table 18 shows the occurrences of strategies for both models on both datasets. Comparing GLM-4.7 to DEEPSEEK-V3.2 (Table 3), we observe that: (i) across models and datasets, the more concise CoT traces contain fewer strategy tags than the reasoning traces; and (ii) the average ratios between CoT and reasoning are of similar magnitude for both models, except for RECON, ERR_SIM, and PLAUS, where the ratio between CoT and reasoning is significantly larger for DEEPSEEK-V3.2 than for GLM-4.7, and CURATE, for which the opposite trend holds.

B.3 Strategies Over Time

Results are shown in Fig. 3 for DEEPSEEK-V3.2 in the CoT setting, and in Fig. 4 and Fig. 5 for GLM-4.7 in the CoT and reasoning settings respectively. Overall trends match those of DEEPSEEK-V3.2 in the reasoning setting (Fig. 2). One notable difference is that GLM-4.7 in CoT consistently engages in ERR_DESC and ERR_SIM, whereas for DEEPSEEK-V3.2 the relative share of these strategies increases over time while CORR and INTER decrease.

B.4 Common Sequences of Strategies

Fig. 6 shows common strategy transitions for DEEPSEEK-V3.2 in the CoT setting, while Fig. 7 and Fig. 8 show the common transitions for GLM-4.7 in The CoT and reasoning settings respectively. Across models and settings, self-transitions for INTER and CORR are frequent, suggesting that these strategies often span multiple segments.

To compare two settings (i.e., models or prompting settings), we compute an agreement score over the major dominant outgoing transitions: for each source strategy and step, we identify the top

Model	Prompt	Strategy	Eedi			SciQ		
			#	Precision	Recall	#	Precision	Recall
DS-V3.2	CoT	Task Interpretation	13	1.00	1.00	24	1.00	1.00
		Correct Answer Ref.	13	1.00	1.00	23	1.00	1.00
		Conceptual Link	0	–	–	15	0.93	0.93
		Error Description	48	0.98	0.89	11	0.64	0.78
		Error Simulation	9	1.00	1.00	0	–	–
		Outcome Instantiation	50	1.00	0.94	66	1.00	0.97
		Plausibility Check	4	0.75	1.00	29	1.00	0.97
		Discriminability Check	2	1.00	1.00	30	0.97	0.97
		Final Set Curation	5	1.00	0.71	7	1.00	0.88
		Reconsideration	1	1.00	1.00	1	1.00	0.50
	Reasoning	Task Interpretation	63	0.87	1.00	43	0.91	1.00
		Correct Answer Ref.	88	1.00	0.96	54	1.00	1.00
		Conceptual Link	0	–	–	29	0.72	1.00
		Error Description	273	1.00	0.99	6	1.00	0.86
		Error Simulation	71	0.99	0.97	0	–	–
		Outcome Instantiation	352	1.00	0.99	176	1.00	1.00
		Plausibility Check	94	1.00	0.98	38	1.00	0.95
		Discriminability Check	43	0.88	1.00	47	0.96	0.83
		Final Set Curation	28	1.00	1.00	37	1.00	0.97
		Reconsideration	46	1.00	0.87	4	1.00	0.80
GLM-4.7	CoT	Task Interpretation	7	1.00	1.00	12	0.83	1.00
		Correct Answer Ref.	17	1.00	0.94	14	1.00	1.00
		Conceptual Link	0	–	–	2	1.00	0.50
		Error Description	103	1.00	0.98	8	1.00	1.00
		Error Simulation	28	1.00	1.00	0	–	–
		Outcome Instantiation	88	1.00	0.98	47	1.00	0.85
		Plausibility Check	12	1.00	1.00	33	0.94	0.97
		Discriminability Check	5	1.00	1.00	25	0.96	0.89
		Final Set Curation	3	1.00	0.75	6	1.00	0.75
		Reconsideration	14	1.00	0.88	0	–	–
	Reasoning	Task Interpretation	19	0.84	1.00	32	1.00	1.00
		Correct Answer Ref.	27	1.00	1.00	18	1.00	1.00
		Conceptual Link	0	–	–	19	0.95	1.00
		Error Description	164	1.00	0.98	12	0.83	0.77
		Error Simulation	49	1.00	1.00	0	–	–
		Outcome Instantiation	241	1.00	0.99	279	1.00	1.00
		Plausibility Check	29	1.00	0.85	72	1.00	0.92
		Discriminability Check	6	0.83	1.00	75	1.00	0.87
		Final Set Curation	16	1.00	0.94	59	0.95	0.93
		Reconsideration	17	1.00	1.00	14	1.00	1.00

Table 14: Performance of LLM-based annotations per taxonomy strategy (Table 1) for DEEPSEEK-V3.2 (DS-V3.2) and GLM-4.7, with CoT vs. reasoning, for both datasets. Note that RECON and CURATE have very little support in the CoT setting.

Strategy	Example 1	Example 2
INTER	“The question asks who is correct regarding the method to find 40% of an amount.”	“The question likely means: among these two numbers, which one(s) are rational?”
CORR	“The correct answer is $\frac{9}{20}$.”	“Remember, 0.5841 rounded to one significant figure [...] gives 0.6.”
LINK	“[...] the options should be animal names.”	“These should be related to bodily movements, digestion, or similar terms [...]”
ERR_DESC	“A student might think $5 + x$ is the same as $5x$.”	“Maybe they simplify $\frac{6}{27}$ to $\frac{1}{3}$?”
ERR_SIM	“If she adds 1 to both sides (incorrectly): $\frac{5x}{3} = 3$.”	“But if you think addition comes before subtraction, you might do $8 + 5 = 13$, then $9 - 13 = -4$.”
INST	“Distractor3: Both Tom and Katie are correct”	“That gives 78,125”
PLAUS	“These are plausible mistakes where a student might [...]”	“This is a plausible distractor if [...]”
DISCR	“‘Cannot be simplified’ is an option, but it is incorrect because it can be simplified.”	“3.7150 is essentially 3.715. So maybe not.”
CURATE	“[...] This is the most robust set.”	“[...] I’ll go with these three.”
RECON	“Wait, is 40 too obvious?”	“But wait: Could there be other possibilities?”

Table 15: Two representative excerpts illustrating each taxonomy strategy (Table 1), taken verbatim (up to deletions marked [...]) from DEEPSEEK-V3.2 and GLM-4.7 reasoning traces. All rows are from Eedi except LINK, illustrated on SciQ where it is prevalent.

Problem (correct answer)	Distractors: gold vs. model	Trace excerpt / failure mode
<i>High-quality (proportional match > 0.5): at least two of three distractors match</i>		
Simplify $\frac{6+9m}{3}$ (correct: $2 + 3m$)	Gold: $6 + 3m$, $2 + 9m$, None of the above Model: $2 + 9m$, $6 + 3m$, $5m$	“[...] $3(2 + 9m)/3 = 2 + 9m$ ”
Median of 3, 5, 6, 18, 18 (correct: 6)	Gold: 10, 12, 18 Model: 10, 18, 5.5	“Confusing median with mean [...] The mean is $(3 + 5 + 6 + 18 + 18)/5 = 50/5 = 10$ ”
30 cl = ★ litres (correct: 0.3)	Gold: 3000, 3, 0.03 Model: 0.03, 3, 30	“[...] some might think [centi] means 1/1000 (like milli) [...]”
<i>Low-quality (proportional match < 0.5): at most one of three distractors matches</i>		
$2\sqrt{5} + 2\sqrt{45} = d\sqrt{5}$ (correct: $d = 8$)	Gold: 7, 20, 4 Model: 5, 6, 12	Solution recovery: the model never reaches $d = 8$, so its values are ungrounded.
$232 \div 5$, remainder as a decimal (correct: 0.4)	Gold: 0.2, 2, 0.25 Model: 46, 4, 0.4	Discriminability: 0.4 is the correct answer, offered as a distractor.

Table 16: High- vs. low-quality generated distractors from DEEPSEEK-V3.2/GLM-4.7 reasoning traces on Eedi, grouped by proportional match (> 0.5 vs. < 0.5, the split used throughout the paper). Each row stacks the human-authored gold distractors over the model’s three proposed distractors, and gives either the verbatim trace excerpt that produced the matching candidate (high-quality; up to deletions [...]) or the failure mode it exhibits (low-quality; §4.3).

Model	Prompt	Eedi		SciQ	
		#correct	#repetitions	#correct	#repetitions
DEEPSEEK-V3.2	Direct	0.42 ± 0.05	0.19 ± 0.06	0.06 ± 0.02	0.00 ± 0.00
	CoT	0.07 ± 0.02	0.02 ± 0.01	0.03 ± 0.02	0.01 ± 0.01
	Reasoning	0.04 ± 0.02	0.02 ± 0.01	0.02 ± 0.01	0.02 ± 0.01
GLM-4.7	Direct	0.57 ± 0.05	0.02 ± 0.01	0.15 ± 0.03	0.01 ± 0.01
	CoT	0.06 ± 0.02	0.03 ± 0.02	0.02 ± 0.01	0.02 ± 0.01
	Reasoning	0.04 ± 0.02	0.02 ± 0.02	0.02 ± 0.01	0.02 ± 0.01

Table 17: Number of correct and repetitions (mean $\pm$ 95% CI based on t-distribution) for DEEPSEEK-V3.2 and GLM-4.7, on the Eedi and SciQ datasets, under different prompting conditions.

Model	Strategy	CoT		Reasoning	
		Eedi	SciQ	Eedi	SciQ
DEEPSEEK-V3.2	Task Interpretation	2.08 ± 0.33	2.45 ± 0.26	11.22 ± 1.38	5.84 ± 0.80
	Correct Answer Ref.	3.27 ± 0.87	1.87 ± 0.22	9.33 ± 1.40	5.52 ± 0.89
	Conceptual Link	0.01 ± 0.02	2.21 ± 0.34	0.04 ± 0.05	5.36 ± 0.87
	Error Description	6.93 ± 1.08	0.56 ± 0.21	22.83 ± 2.70	0.78 ± 0.31
	Error Simulation	2.09 ± 0.62	0.00 ± 0.00	5.70 ± 1.18	0.00 ± 0.00
	Outcome Instantiation	8.03 ± 1.32	8.10 ± 1.17	34.84 ± 4.05	24.71 ± 2.95
	Plausibility Check	1.26 ± 0.31	2.72 ± 0.41	7.94 ± 1.15	5.33 ± 0.90
	Discriminability Check	0.51 ± 0.25	3.18 ± 0.53	2.54 ± 0.59	10.34 ± 1.50
	Final Set Curation	0.85 ± 0.17	1.42 ± 0.26	3.34 ± 0.39	5.83 ± 0.60
	Reconsideration	0.67 ± 0.22	0.16 ± 0.11	5.46 ± 0.71	1.05 ± 0.23
GLM-4.7	Task Interpretation	1.16 ± 0.18	1.89 ± 0.15	5.31 ± 1.31	4.18 ± 0.34
	Correct Answer Ref.	4.99 ± 3.45	1.67 ± 0.17	9.39 ± 3.09	3.72 ± 0.55
	Conceptual Link	0.01 ± 0.01	0.79 ± 0.27	0.03 ± 0.04	1.86 ± 0.43
	Error Description	24.68 ± 32.08	1.22 ± 0.32	27.45 ± 9.20	2.34 ± 0.44
	Error Simulation	14.39 ± 20.74	0.00 ± 0.00	14.41 ± 7.86	0.00 ± 0.00
	Outcome Instantiation	8.09 ± 1.80	8.40 ± 1.52	34.97 ± 10.28	39.54 ± 4.64
	Plausibility Check	1.18 ± 0.37	4.65 ± 0.50	4.58 ± 0.91	10.32 ± 1.42
	Discriminability Check	0.21 ± 0.13	4.67 ± 0.69	0.96 ± 0.32	10.20 ± 1.60
	Final Set Curation	0.56 ± 0.21	1.30 ± 0.30	3.96 ± 0.63	7.20 ± 1.14
	Reconsideration	0.94 ± 0.34	0.10 ± 0.10	2.76 ± 0.70	1.89 ± 0.43

Table 18: Average occurrences of taxonomy strategies (Table 1) in CoT and reasoning traces for each model on both datasets (mean ± 95% CI based on t-distribution).

three transitions exceeding 15% outgoing probability mass and compute the set overlap between DEEPSEEK-V3.2 and GLM-4.7, normalized by the smaller set size. Averaging over source strategies and steps yields an agreement of 0.90 when varying only models (and averaging over datasets), and 0.91 when varying only prompting settings (and averaging over models), indicating broadly consistent dominant transition structures. This agreement score still permits several minor differences between settings: (i) the ordering of outgoing dominant edges may differ; for example, on Eedi, the most dominant outgoing edge of GLM-4.7 in CoT from ERR_DESC is to ERR_SIM, whereas for DEEPSEEK-V3.2 it is to ERR_DESC. The former corresponds to iteratively listing an error and directly simulating it, whereas the latter corresponds to listing multiple errors at once. (ii) even when the ordering of dominant edges is consistent, the ratios between transition masses can differ; for instance on Eedi, for INST in CoT, the ratio of transitions to ERR_DESC vs self-transitions is much larger for GLM-4.7 than for DEEPSEEK-V3.2. (iii) the number of dominant edges may vary; for example on Eedi, CORR has 4 dominant outgoing edges in DEEPSEEK-V3.2’s reasoning traces compared to just 2 in GLM-4.7’s.

C Details on Discussion Experiments

This section provides a detailed description of the experiments performed as part of the discussion (§4.3).

C.1 Solve-First and Error Injection

To measure how frequently the LLM first solves a problem correctly and then injects an error into the solution, we annotate the LLMs’ traces at scale. Following the approach used in our main analysis, we first define the following taxonomy:

- SOLVE_FIRST: Does the model systematically solve the problem to reach the correct final answer?
- INJECT_FROM_SOLUTION: Does the model refer to the correct solution or intermediate steps to generate plausible errors, misconceptions, or distractors?

We limit the analysis to problems where the LLM generated at least two steps in a step-by-step solution, ensuring that error injection is possible. Step-by-step solutions are generated using the prompt in Table 19. Validation is performed by measuring the agreement with manual annotation on 50 traces. This yields a Cohen’s κ of 0.85 ($p < 0.001$). To

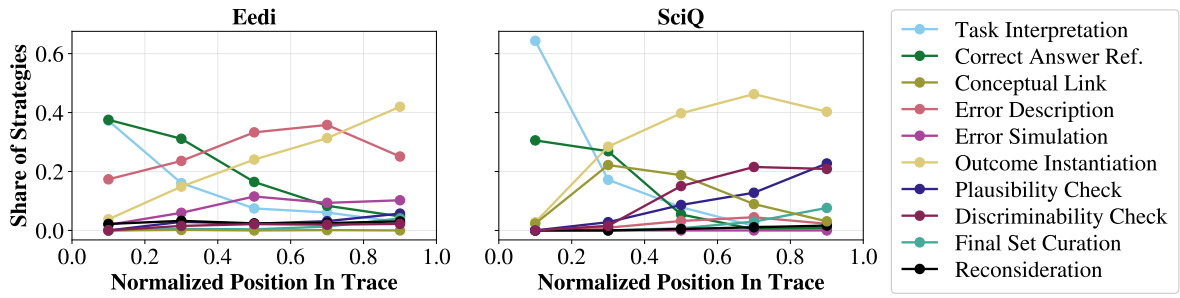


Figure 3: Indicates how often each strategy in our taxonomy Table 1 was annotated at different stages of DEEPSEEK-V3.2's CoT trace. Time is normalized (0 = start of trace, 1 = end). Note that the proportions sum to one for each of the five temporal bins.

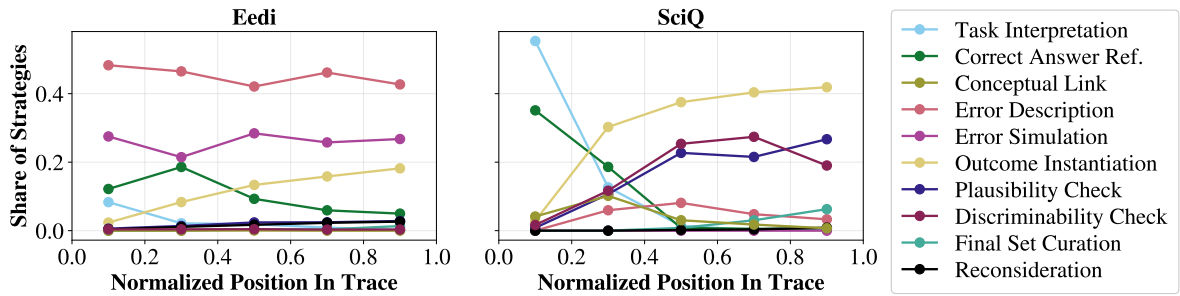


Figure 4: Indicates how often each strategy in our taxonomy Table 1 was annotated at different stages of GLM-4.7's CoT trace. Time is normalized (0 = start of trace, 1 = end). Note that the proportions sum to one for each of the five temporal bins.

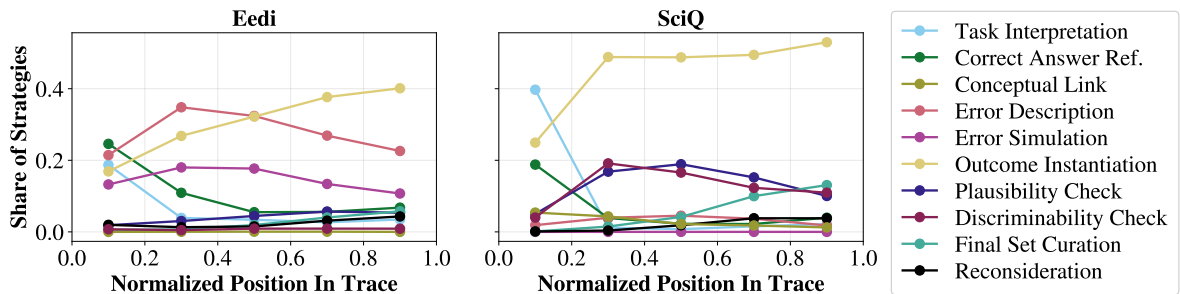


Figure 5: Indicates how often each strategy in our taxonomy Table 1 was annotated at different stages of GLM-4.7's reasoning trace. Time is normalized (0 = start of trace, 1 = end). Note that the proportions sum to one for each of the five temporal bins.

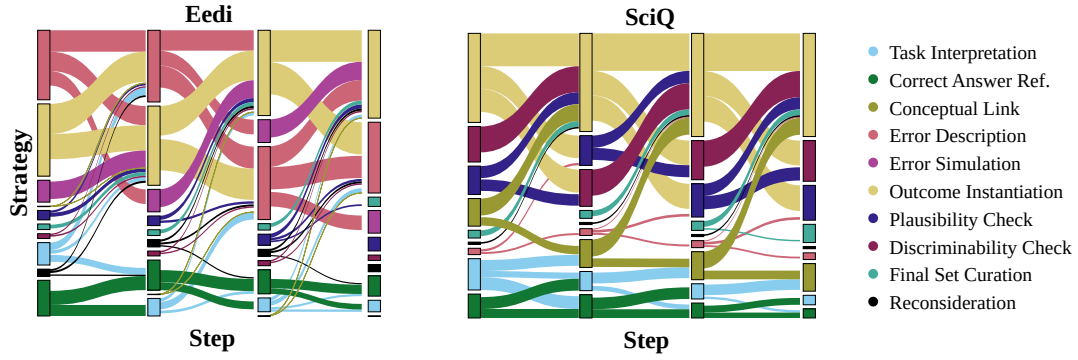


Figure 6: Shows transition probabilities between strategies in DEEPSEEK-V3.2’s CoT traces. Sequences of strategies up to length 4 (left to right). Node height represents strategy share and link widths indicate the transition probabilities between successive strategies. Only dominant (>15% outgoing mass) transitions are visualized for simplicity.

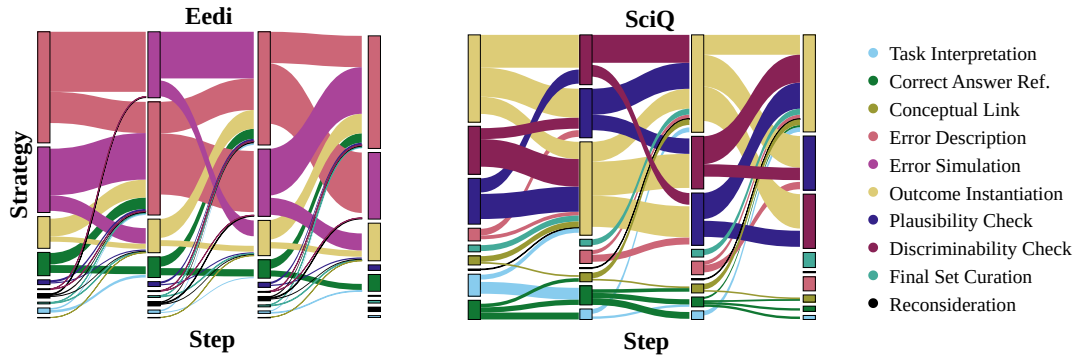


Figure 7: Shows transition probabilities between strategies in GLM-4.7’s CoT traces. Sequences of strategies up to length 4 (left to right). Node height represents strategy share and link widths indicate the transition probabilities between successive strategies. Only dominant (>15% outgoing mass) transitions are visualized for simplicity.

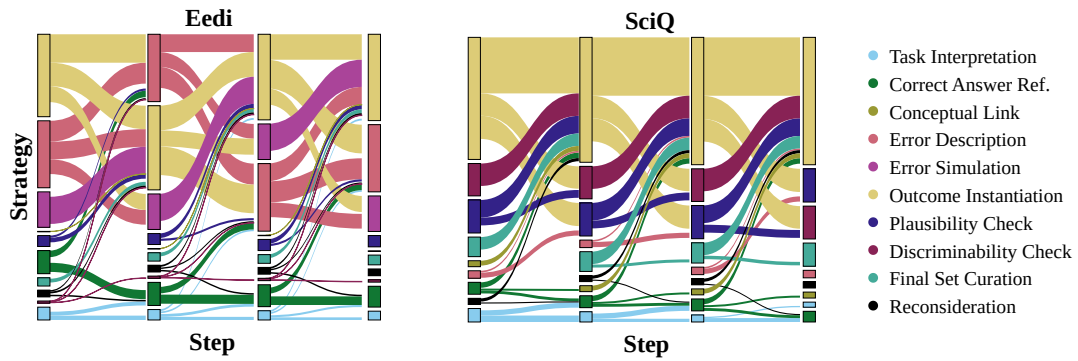


Figure 8: Shows transition probabilities between strategies in GLM-4.7’s reasoning traces. Sequences of strategies up to length 4 (left to right). Node height represents strategy share and link widths indicate the transition probabilities between successive strategies. Only dominant (>15% outgoing mass) transitions are visualized for simplicity.

identify similarity-based strategies, we manually inspect reasoning traces filtered for semantically similar distractor values.

We find that in 92.5% of DEEPSEEK-V3.2 traces and 97.8% of GLM-4.7 traces, the model constructs a full step-by-step solution before proposing distractors. In 71.2% (DEEPSEEK-V3.2) and 76.3% (GLM-4.7) of these cases, the model diverges at a specific step to inject an error, closely mirroring misconception-based design.

Automatic annotation. GPT-5.2 (OpenAI Team, 2026) is used to automatically annotate both SOLVE_FIRST and INJECT_FROM_SOLUTION for all traces. Explanations or relevant excerpts from the distractor-generation trace are also extracted to support the assigned labels (prompt in Table 20).

Annotation quality. A stratified subset of 50 samples are labeled manually. Agreement between manual and automatic annotations yields $\kappa = 0.85$ ($p < 0.001$).

System Prompt

You are a helpful assistant.

User Prompt

Solve the math problem. Output format (exact): [STEP-1]... ... [STEP-N]... [FINAL ANSWER]... Constraints: no fluff; only one reasoning/arithmetic step at a time; show key work. Final answer MUST equal: "{final_answer}" Problem: "{problem_formulation}"

Table 19: Prompt for solving a math problem $\{problem_formulation\}$ step-by-step, in a manner that is consistent with the ground truth answer $\{final_answer\}$.

System Prompt

You will be given a reasoning trace whose purpose is to generate *plausible incorrect distractor answers* for a math problem (i.e., simulate student mistakes). Your job is to label two independent properties: **1. SOLVE_FIRST:** Does the trace *attempt to solve the original problem correctly step-by-step* (compute the correct answer path), even if it later discusses mistakes/distractors? - YES if it tries to lay out a solution procedure and/or computes intermediate results toward the true answer. - NO if it jumps straight to distractors/misconceptions without first doing a solution attempt. **2. INJECT_FROM_SOLUTION:** Does the trace *use the correct solution path as scaffolding* by referencing a specific step/intermediate quantity from the correct solution and then “branching off” by applying an incorrect operation/value there, propagating that error to a final distractor? - YES only if you can point to a concrete “branch point” tied to the correct solution (e.g., “Up to step X it’s correct; then it miscomputes X or applies the wrong operation to X, yielding distractor Y”). - NO if distractors are proposed without an explicit branch point from the correct solution (e.g., “a possible distractor is 30 m/s” with no tie to a specific intermediate step), OR if it only lists misconceptions abstractly without anchoring them to a step in the correct solution. These labels are **independent** : A can be YES while B is NO (solves correctly first, then proposes distractors without branching). B is unlikely if A is NO, but still label based on the text. Return output in EXACTLY this format (for regex parsing): SOLVE_FIRST_DISCUSSION: <short quote or paraphrase from trace that is relevant for SOLVE_FIRST; or “N/A”> SOLVE_FIRST_LABEL: YES NO INJECT_FROM_SOLUTION_DISCUSSION: <short quote or paraphrase from trace that is relevant for INJECT_FROM_SOLUTION; or “N/A”> INJECT_FROM_SOLUTION_LABEL: YES NO **Mini-examples (for guidance only):** Example 1 (SOLVE_FIRST=YES, INJECT_FROM_SOLUTION=YES): Reasoning Trace: “Let’s solve: $30\text{min}=0.5\text{h}$; $v=60/0.5=120\text{km/h}$; $/3.6=33.3\text{m/s}$. Student might incorrectly compute $60/0.5$ as 30km/h , then convert...” Expected Output: “SOLVE_FIRST_DISCUSSION: Correct problem solving present. SOLVE_FIRST_LABEL: YES INJECT_FROM_SOLUTION_DISCUSSION: Branches off at the $60/0.5$ step with a calculation mistake. INJECT_FROM_SOLUTION_LABEL: YES” Example 2 (SOLVE_FIRST=NO, INJECT_FROM_SOLUTION=NO): Reasoning Trace: “A potential distractor is 30 m/s .” Expected Output: “SOLVE_FIRST_DISCUSSION: No problem solving attempted. SOLVE_FIRST_LABEL: NO INJECT_FROM_SOLUTION_DISCUSSION: N/A INJECT_FROM_SOLUTION_LABEL: NO” Example 3 (SOLVE_FIRST=YES, INJECT_FROM_SOLUTION=NO): Reasoning Trace: “The correct solution is: $30\text{min}=0.5\text{h}$; $v=60/0.5=120\text{km/h}$; $/3.6=33.3\text{m/s}$. One potential distractor is 45m/s .”

Expected Output:
“SOLVE_FIRST_DISCUSSION: Question is first solved correctly. SOLVE_FIRST_LABEL: YES INJECT_FROM_SOLUTION_DISCUSSION: No solution step is referenced and no related branching off takes place. INJECT_FROM_SOLUTION_LABEL: NO” Now label the following reasoning trace. [BEGIN TRACE] {trace} [END TRACE]

Table 20: Prompt for annotating solve-first and inject-from-solution behavior in LLMs’ traces $\{trace\}$ when generating distractors.

C.2 Error Simulation

Simulation prompt. Errors are simulated using the prompt in Table 21, with DEEPSEEK-V3.2 in reasoning mode.

System Prompt

You will be given a math question and specific student error. Please generate the incorrect answer that a student would give on the current question if they made the specified error. At the end, give the student’s final concise answer preceded with ‘Incorrect Student Answer:’

User Prompt

Question: {problem_formulation} Student Error: {error}

Table 21: Prompt for generating an incorrect student answer conditioned on a specific misconception $\{error\}$ on one math word problem $\{problem_formulation\}$.

Manual analysis for consistency. To evaluate consistency with any reasonable interpretation of the provided error, we first compare the generated answer against all available distractors for the same question that are sourced from the same error. For cases where the answer does not match any available distractor, we inspect the reasoning trace step-by-step to verify whether the error was correctly implemented. Albeit rarely the case, we consider examples consistent if the error is correctly implemented but is combined with a second error. Most instances are straightforward to evaluate; for example, errors such as “rounds too much” are trivial to classify.

Trace level analysis. We analyze the reasoning traces for whether they contain the correct step by step solution to the problem, whether the misconception is discussed right in the beginning, after the correct solution, or injected midway at the relevant step, the dominant stance used by the LLM (student vs. expert), etc. We annotate these aspects for each trace using GPT-4.1 (see Table 22 for the prompt used), and perform a manual verification of the annotations on a subset of 20 traces

(Cohen’s κ : 0.804 for application_pattern, 0.894 for correct_path_present, 0.898 for dominant_stance, and 1.0 for the rest of the tags, $p < 0.001$ for all tags). We find that 68.3% traces contained the correct solution to the underlying problem. 64.34% traces first produced the correct solution, then introduced the misconception, whereas 30.3% discuss applying the misconception from the start. DEEPSEEK-V3.2 held a consistent expert stance in almost all (99.3%) traces (of which, 66.9% third person and 32.4% first person). We also found that valid traces, i.e., resulting in the desired misconception-grounded distractor, were significantly shorter (based on a Mann-Whitney U test; U : 5728, $p < 0.05$) in terms of average step count (3.377 steps) compared to invalid traces (3.972 steps).

System Prompt

You are analyzing a chain-of-thought trace produced by an LLM that was asked to simulate a wrong answer a student would give to a math question, conditioned on a specific student misconception. Your job is to characterize *how* the trace handles the misconception – not to grade the math. You will be given: - question: the math question shown to the simulator
- correct_answer: the ground-truth answer
- misconception: the high-level error the simulator was told to apply
- sim_answer: the distractor the simulator produced
- sim_reasoning: the simulator’s full chain-of-thought
Read sim_reasoning carefully and assign the following labels.

correct_path_present
True iff the trace explicitly derives or states the correct answer at any point, independent of the misconception. Mentioning the correct method without computing it counts as False.

application_pattern
The dominant structural shape of the trace.
- correct_then_corrupt_full : correct answer derived in full, then misconception introduced and the erroneous path re-executed end-to-end
- correct_then_corrupt_skip : correct answer derived in full, then misconception introduced and the final wrong answer stated without re-executing the steps
- corrupt_partial : correct path begun, misconception introduced mid-derivation, execution continues from that point with the corrupted step
- corrupt_first : misconception applied from the start; correct path absent or only acknowledged in passing
- other **misconception_evidence**
1-3 short verbatim snippets from sim_reasoning that justify the labels above in relation to where and how the misconception is induced in sim_reasoning; each ≤ 25 words.

dominant_stance
The overall narrative perspective of the trace.
- expert_third_person : the model speaks *about* the student in the third person (“the student would do X”, “they might think Y”). Pedagogical / diagnostic framing.
- first_person_student : the model speaks *as* the student, in the first person (“I think...”, “I’ll multiply...”, “my answer is...”), inhabiting the role.
- first_person_solver : the model uses first-person language but as itself solving / deciding (“I need to compute...”, “I’ll go with 1 as the answer”), not roleplaying the student. This is the AI narrating its own task execution.
- impersonal : neither perspective is marked – the trace is mostly bare computation, passive voice, or pronoun-free statements (“ $4 \times -3 = -12$. Therefore the answer is -12.”).
- mixed : two or more stances are roughly comparable in prominence and neither dominates.

stance_shift
Does the stance change across the trace?

- stable : one stance throughout
- shifts_to_student : starts expert/solver/impersonal, moves into first-person student voice at some point
- shifts_from_student : starts in student voice, moves out of it
- oscillates : repeatedly switches between stances
- n_a : trace is too short or too uniform to assess

student_voice_present
True iff the trace contains at least one clear instance of the model speaking *as* the student in the first person (not merely first-person solver language).

expert_voice_present
True iff the trace contains at least one clear instance of expert/third-person framing about the student (“the student would...”, “they might...”, “a learner who believes X...”).

pedagogical_framing
True iff the trace explicitly reasons *about* the misconception as a teaching / diagnostic object (e.g. naming it, classifying it, discussing what kind of student holds it, comparing it to other errors) rather than just applying it.

stance_evidence
2-4 short verbatim snippets from sim_raw or sim_reasoning that justify the labels above in relation to the stance used; each ≤ 25 words. Include at least one snippet for each stance you marked as present.

User Prompt

question: {problem_formulation}
correct_answer: {correct_answer}
misconception: {student_error}
sim_answer: {sim_answer}
sim_reasoning: {sim_reasoning}

Table 22: Prompt for annotating error simulation reasoning traces.

C.3 Enforcing a Literature-Informed Reasoning Structure

Literature-informed prompt. To test whether models benefit from being explicitly instructed to follow the misconception-based procedure, we use the prompt in Table 23, which enforces the full reasoning structure: solve the problem first, enumerate common error primitives, simulate each error, assess plausibility and discriminability, and select the final set. Table 24 reports proportional match under this literature-informed prompt (w/ *LS*) against the neutral prompt (w/o *LS*).

System Prompt

You will be given a math question. Please generate {n} incorrect distractor answers for the question to be used as multiple-choice options in a multiple-choice exam.

****RULES****

Solve correctly first:
1. First, solve the problem correctly and treat the most likely correct solution as fixed.
2. Identify the key concepts involved in the correct solution.
3. Specify the exact conditions that an answer must violate to be a valid distractor.

Error modeling:
4. Enumerate at most 7 common error primitives relevant to this problem. Each error primitive must be either:
(a) a buggy rule (an incorrect transformation or procedure), or
(b) a buggy commitment (a false assumption or misclassification treated as true).
5. Each error primitive must be specific, stable, and capable of producing a concrete deterministic answer.

Error simulation:

```

6. For each error primitive, assume the student fully
   commits to that single error and reasons correctly in all
   other respects.
7. Derive the final incorrect answer that results from that
   single error and collect it as a distractor candidate.
   - If the question is non-numerical, output the final
   incorrect conclusion, classification or choice that follows
   from the error primitive.
Plausibility Assessment:
8. For each distractor candidate you assess its
   discriminative power:
   - check if the candidate can be derived deterministically
   - check if the candidate is truly incorrect under any
   reasonable interpretation
   - quantify how likely students are to stop here and select
   this candidate
   - make sure it is unambiguous and well-formed
Selection:
9. From the remaining candidates, select the {n} most
   distinct plausible distractors.
Output rules:
10. You should only output the final concise distractor
   values in the following template:
[Template]
Distractor1:
...
Distractor{n}:
-----
User Prompt
-----
Question: {problem_formulation}

```

Table 23: Prompt for enforcing an ideal, learning-science-informed distractor generation procedure for a single math problem *{problem_formulation}*.

The two annotated models. For DEEPSEEK-V3.2 and GLM-4.7, enforcing the structure yields no significant change ($p > 0.05$, paired t -test; Table 24), consistent with our interpretation that they already employ a process closely resembling the misconception-based approach in their reasoning (§4.3).

Generalization across models. The process-level analysis covers only DEEPSEEK-V3.2 and GLM-4.7, as validating the automatic annotator is costly (discussed in limitations §5). The *performance* measurements, however, need no annotation, so we can report proportional match for a wider set of reasoning models and test whether the performance-level findings generalize. Beyond DEEPSEEK-V3.2 and GLM-4.7 we evaluate GPT-OSS-20B, GPT-OSS-120B, GPT-5, GLM-4.7 FLASH, GEMMA-3, and GEMINI-2.5 (FLASH/PRO), all in reasoning mode, on the same 429 solvable Eedi problems.

The clearest generalization is the strong absolute gap between math and science: on SciQ under the neutral prompt, proportional match stays in the 0.11–0.15 band for every model we ran there (GPT-OSS-20B 0.11, GPT-OSS-120B 0.12, GLM-4.7 FLASH 0.12, DEEPSEEK-V3.2 0.14, GLM-4.7 0.15), far below the Eedi range, mirroring the Eedi/SciQ gap we analyze. Absolute Eedi performance

itself varies across models, as expected for models of different families and sizes (see Table 24).

Enforcing the reasoning structure, by contrast, does not generalize uniformly. It produces no significant change for DEEPSEEK-V3.2, GLM-4.7, GEMMA-3, and GEMINI-2.5 (FLASH/PRO) ($p > 0.05$, paired t -test), consistent with our claim that these models already self-organize an effective reasoning structure (§4.3). The models that improve are all from OpenAI: GPT-OSS-20B, GPT-OSS-120B, and GPT-5 (all $p < 0.01$; Table 24). These are also the lowest-scoring models under the neutral prompt, so model family and performance headroom are confounded and we do not attribute the gain to either. None of this bears on our claim, which concerns only the two models we annotate in depth: neither improves under enforced structure, consistent with the annotation finding that their traces already contain the misconception-based steps (§4.3). GLM-4.7 FLASH is the only model for which the neutral prompt scores significantly higher, and even there the effect is borderline. Under the structure-enforcing prompt it frequently falls into repetition loops, so we attribute its drop to a failure to follow the multi-step instruction rather than to a cost of the structure itself.

C.4 Oracle Ranker over Proposed Candidates

The oracle ranker isolates how much of the selection loss (§4.3) comes from the model discarding a valid candidate rather than never proposing it. We count a human-authored distractor as recovered if the model instantiates an equivalent value anywhere in its reasoning, not only in its final three.

Oracle proportional match. For each problem we build a candidate pool: the model’s final selected distractors, together with every candidate it proposes elsewhere in its reasoning (recovered from the INST spans; details below). We call a human-authored distractor *matched* if some pool member is judged equivalent to it, using the same GPT-4.1 judge as in §A.2. The oracle proportional match is the fraction of human-authored distractors matched this way, counting each at most once. The actual proportional match is the same fraction over the final selected distractors alone. The oracle is therefore always at least the actual, and their difference counts the distractors the model proposed but did not select.

Recovering proposed candidates from INST.

To recover the proposed candidates we use the INST

Model	Size	w/o LS	w/ LS
DEEPSEEK-V3.2	685b	0.52 ± 0.03	0.55 ± 0.03
GLM-4.7	358b	0.52 ± 0.03	0.52 ± 0.03
GPT-OSS-20B	20b	0.30 ± 0.03	0.35 ± 0.03
GPT-OSS-120B	120b	0.36 ± 0.03	0.46 ± 0.03
GPT-5	–	0.47 ± 0.03	0.52 ± 0.03
GLM-4.7 FLASH	31b	0.40 ± 0.03	0.37 ± 0.03
GEMMA-3	31b	0.57 ± 0.03	0.55 ± 0.03
Gemini-2.5-flash	–	0.53 ± 0.03	0.54 ± 0.03
Gemini-2.5-pro	–	0.52 ± 0.03	0.53 ± 0.03

Table 24: Proportional match on the 429 solvable Eedi problems (reasoning mode) under the neutral prompt (*w/o LS*) and the literature-informed prompt enforcing the full reasoning structure (*w/ LS*). The two models we annotate in depth (top block) show no significant change from enforced structure; among the additional models, only the OpenAI models (GPT-OSS-20B, GPT-OSS-120B, and GPT-5) improve significantly. These improvers are also among the lowest-scoring models under the neutral prompt, so a model-family effect and a performance-headroom effect are confounded here and we do not attribute the gain to either. **Bold** marks the setting that is significantly better per model (paired *t*-test, $p < 0.05$); non-bold pairs do not differ significantly.

tags, which mark where the model instantiates a concrete candidate distractor. For every INST occurrence we reconstruct a short excerpt of the 220 characters preceding the tag, so that the excerpt terminates exactly at the proposed value. We then prompt GPT-4.1 (greedy decoding) to extract the concrete answer value at the end of the excerpt (Table 25); the question and correct answer are supplied for context. We validated the extraction on 100 held-out excerpts sampled uniformly across datasets, models and CoT vs reasoning setting. We manually judge each extracted value against the excerpt: 98% were correct.

Population estimates via inverse-probability weighting. Because the annotated traces are stratified on performance (§C.6), a raw mean over them does not estimate the population proportional match. To report population-level figures we inverse-probability-weight each trace by the inverse of its within-bucket sampling rate, the population count of its bucket divided by the number of traces sampled from it. The two buckets are solvable traces with proportional match > 0.5 and < 0.5 . For DEEPSEEK-V3.2 this recovers the reported full-data proportional match: 0.51 on Eedi, within the reported 0.52 ± 0.03 , and 0.14 on SciQ, matching the reported value exactly. Table 26 reports the population-weighted oracle and actual proportional match for all eight dataset $\times$ model $\times$ prompting-setting combinations. Note how the paired *t*-test needs no weighting because it is a within-trace comparison: each trace

contributes its own oracle-minus-actual difference, and the test asks only whether that difference is reliably positive. Weighting matters solely for projecting the stratified sample onto a population average (the point estimates), not for a difference evaluated separately within each trace, where the oracle is by construction never below the actual.

System Prompt
You are given an excerpt from an LLM’s reasoning while it invents wrong answer options (distractors) for a multiple-choice question. The excerpt ends with a <INST> marker. Immediately to the LEFT of that marker is one candidate distractor value that the reasoning just named. Your job is purely positional: output the value token(s) directly before <INST>, not any other value appearing earlier in the excerpt. Clean it up: strip surrounding markdown/formatting, quotes, enumeration or field labels (e.g. ‘Distractor1:’, ‘Idea 3:’, ‘3.’, ‘F =’), and any trailing bracketed or parenthetical comment. Use the question and correct answer only to judge where the value begins. Output ONLY that value, or exactly NONE if there is no concrete answer value immediately before the marker.
User Prompt
<pre><question>{question}</question> <correct_answer>{correct answer}</correct_answer> <excerpt>{~220-char excerpt ending at the candidate}<INST></excerpt></pre>

Table 25: Prompt used to extract the concrete candidate distractor value marked by each INST span (§C.4). The <INST> marker is re-appended at the end of the excerpt so the extractor targets the value the tag annotates rather than an earlier-mentioned candidate.

C.5 Valid Distractors Dropped by Self-Checks

The oracle analysis (§C.4) shows that models frequently propose a valid distractor and then discard it. Here we ask how much of that loss the model’s own plausibility and discriminability

Dataset	Model	Setting	Actual	Oracle	p
Eedi	DEEPSEEK-V3.2	reasoning	0.51	0.67	7.9×10^{-12}
Eedi	DEEPSEEK-V3.2	CoT	0.51	0.57	1.1×10^{-4}
Eedi	GLM-4.7	reasoning	0.53	0.70	1.5×10^{-11}
Eedi	GLM-4.7	CoT	0.46	0.52	5.5×10^{-5}
SciQ	DEEPSEEK-V3.2	reasoning	0.14	0.29	9.8×10^{-8}
SciQ	DEEPSEEK-V3.2	CoT	0.14	0.19	1.0×10^{-3}
SciQ	GLM-4.7	reasoning	0.19	0.29	5.7×10^{-7}
SciQ	GLM-4.7	CoT	0.16	0.20	2.2×10^{-3}

Table 26: Oracle vs. actual proportional match for all eight dataset×model×prompting-setting combinations. The *oracle* counts a human-authored distractor as recovered whenever the model instantiates an equivalent value anywhere in its reasoning; the *actual* counts only the final selected set. Both cap each human-authored distractor at one match and are inverse-probability-weighted to the population (§C.4); p is the per-trace paired t -test (unaffected by weighting), and n is 120 per Eedi cell and 77–88 per SciQ cell. The oracle exceeds the actual in every cell.

checks (PLAUS, DISCR) account for: we quantify how often such a check argues against keeping a candidate that matches a human-authored one.

Linking checks to candidate and sentiment. To count these cases we must know which candidate each check assesses, so we link every PLAUS/DISCR check to an INST candidate. Our annotations already mark both the check tag and the INST candidates, but not which candidate a given check refers to. Because the tag sits at the end of the assessment and the candidate value can lie before or after it,⁸ we take a window of ~ 300 characters to the check’s left and ~ 160 to its right. Within this window we strip all other tags and replace the check with a <<<CHECK>>> marker, then prompt GPT-4.1 to (i) extract the single INST candidate value the check assesses, and (ii) classify its *sentiment toward keeping that candidate* as positive, negative, or neutral (full prompt in Table 27). Some checks target the distractor set as a whole or make a generic remark; for these the extractor returns NONE, and we exclude them from the drop-rate analysis. Sentiment is *negative* when the check argues against using the candidate, for example when the model judges it indistinguishable from the correct answer, too obvious or too obscure to be tempting, or an invalid answer form; *positive* when it argues for keeping it; and *neutral* when the remark is descriptive or undecided. We validated the joint extractor and classifier against manual judgements on 100 held-out checks (50 PLAUS, 50 DISCR) sampled across datasets, models, and prompting settings, reaching $\sim 93\%$ agreement.

⁸For instance, the value may sit to the left of the tag (“...giving 3600 <INST>”, but this is too close to the correct total <DISCR>”) or to its right (“this is a tempting slip <PLAUS>”, giving 3600 <INST>”).

System Prompt

You are analyzing an excerpt from an LLM inventing wrong answer options (distractors) for a multiple-choice question. The marker <<<CHECK>>> is a {check type} check the model applied. The check’s evaluative clause ENDS at the marker, so read the text immediately to the LEFT of <<<CHECK>>> to understand what the check concludes. Base BOTH the value and the verdict on that left clause; text AFTER the marker is the next reasoning step and must not override it.

STEP 1 – VALUE: identify the single candidate distractor value that this check is assessing. Use the left clause to decide WHICH candidate is meant; the candidate’s VALUE may appear either just before (left of) or just after (right of) the marker – extract it from whichever side it appears. Output it verbatim, cleaned of labels/markup/quotes. Numbers or expressions that are part of the QUESTION itself (given terms, the sequence in the prompt, the correct answer being noted as correct) are NOT the assessed distractor value. OUTPUT NONE (do not pick any single value) when the check does not target exactly ONE candidate, i.e. any of:

- * it evaluates the distractor SET or SEVERAL candidates together – tell-tale words ‘these’, ‘they’, ‘all’, ‘both’, ‘the three’: e.g. ‘these are plausible’, ‘these are reasonable errors’, ‘these are all incorrect’, ‘these are distinct’, ‘all close enough to the correct answer’, ‘these look good’. This applies even when specific values appear earlier in the excerpt: if the clause ENDING AT the marker is a collective judgement – including a list of several values followed by one verdict (‘630, 600 and 700 are plausible’, ‘Incorrect: 48, 190, 112. These are plausible mistakes’) – output NONE, do NOT pick one of them;
- * it is a general/procedural/meta remark (‘I need three distinct distractors’, ‘let me finalize’, ‘so any method that does not yield X is incorrect’);
- * it verifies or restates the CORRECT answer (‘X is correct’, ‘the logic holds’, ‘Tom is correct’).

Only output a value when the check clearly assesses exactly ONE specific candidate distractor.

ALWAYS give a SENTIMENT in STEP 2, even when VALUE is NONE: judge the check’s stance toward the distractor(s) it discusses. A collective endorsement (‘these are plausible’, ‘these are good distractors’, ‘plausible but clearly wrong’) is POSITIVE; a collective rejection (‘these are too obvious’) is NEGATIVE.

Sentiment rule, plausibility variant (PLAUS):

STEP 2 – SENTIMENT: what is this plausibility check’s SENTIMENT toward keeping the distractor?

A good distractor must be TEMPTING – a wrong answer a student might actually pick.

- POSITIVE if the check endorses it: plausible/tempting/believable/a strong distractor, OR names any real reason a student could be tempted (a shared property, a common confusion, ‘sounds like’, ‘related to’, ‘contains X’) – even if the same sentence also notes it is ultimately wrong. A distractor being INCORRECT is expected and is NOT negative.
- NEGATIVE if the check argues against using it: not tempting, too obvious, too far-fetched/obscure, ‘unlikely’, ‘not plausible’, ‘a student who computes would see it’s wrong’, ‘too easy to rule out’, ‘less common’/‘rare’, OR the

```

model compares candidates and leans AGAINST this one
(favours another over it).
- NEUTRAL if purely descriptive or undecided ('but
maybe...').
Sentiment rule, discriminability variant (DISCR):
STEP 2 - SENTIMENT: what is this discriminability check's
SENTIMENT toward keeping the distractor?
A good distractor must be clearly WRONG yet still worth
offering (not secretly correct, not so obvious no one picks
it).
- POSITIVE if the check endorses it as a good distractor:
clearly/unambiguously WRONG yet usable. Being wrong is GOOD
for a distractor, so 'plausible but clearly wrong upon
careful checking', 'not the first/main/primary X', 'the
reverse process', 'not directly Y', or simply the wrong
answer => POSITIVE.
- NEGATIVE if the check argues against using it: it might
actually be CORRECT or is indistinguishable from / equal to
the correct answer ('that's correct, so not a distractor'),
too close/ambiguous, an invalid answer form, so obviously
wrong no one would pick it, OR the model rejects/sets it
aside ('not exactly', 'doesn't fit', 'less common', 'let's
avoid').
- NEUTRAL if purely descriptive or undecided ('but maybe the
question expects...').
Output exactly two lines:
VALUE: <value or NONE>
SENTIMENT: <POSITIVE|NEGATIVE|NEUTRAL>
-----
User Prompt
-----
<question>{question}</question>
<correct_answer>{correct answer}</correct_answer>
<excerpt>{excerpt with <<<CHECK>>> marker}</excerpt>

```

Table 27: Prompt used to (i) extract the single INST candidate value assessed by each PLAUS/DISCR self-check and (ii) classify its sentiment toward keeping that candidate (§C.5). The *{check type}* token and the STEP 2 sentiment rule take the plausibility variant for a PLAUS check and the discriminability variant for a DISCR check; the rest of the prompt is shared.

Drop rates. Each check is now linked to a candidate and a sentiment. We label the candidate *valid* if its value is judged equivalent to some human-authored distractor and *invalid* otherwise.⁹ For each dataset×model×prompting-setting combination we report the fraction of PLAUS/DISCR checks on valid candidates that are negative (the *valid drop rate*), alongside the same fraction over checks on invalid candidates (the *invalid drop rate*). We report rates rather than counts because the model proposes many more invalid than valid candidates. Rates are inverse-probability-weighted to the population using the same per-bucket weights as in §C.4 (each check inherits its trace’s weight). Table 28 shows the results. The valid drop rate is below the invalid drop rate in every setting: for DEEPSEEK-V3.2 reasoning the invalid drop rate is 1.9× the valid rate on Eedi (53% vs. 28%) and 1.5× on SciQ (23% vs. 15%), so the checks discriminate, rejecting a given invalid candidate more often than a given valid one.

⁹Using the same GPT-4.1 judge as for proportional match (§A.2).

This discrimination is only relative, however: in absolute terms a non-trivial share of the checks on valid candidates are still negative (up to 28% on Eedi reasoning). The pattern is consistent across both models: while GLM-4.7’s valid and invalid drop rates are both lower than DEEPSEEK-V3.2’s in every reasoning cell, the ordering (invalid > valid in every cell, Eedi > SciQ, reasoning > CoT) is identical. The CoT rows carry too few checks on valid candidates to be reliable (valid $n \leq 7$), so the cross-model comparison above rests on the reasoning rows. Split by check type, the two are negative at comparable rates (on Eedi reasoning, DEEPSEEK-V3.2: 29% of PLAUS and 27% of DISCR checks on valid candidates). A negative check does not always translate into a drop: the model may keep the candidate anyway, or discard it for unrelated reasons. Of the checks on valid candidates, the share that are negative *and* whose candidate is absent from the final three is a lower bound on the loss these checks drive: 19% on Eedi and 10% on SciQ for DEEPSEEK-V3.2 reasoning.

C.6 Strategy Usage and Performance

To test whether the strategies in our taxonomy are correlated with output quality, we ask, in two stages, whether and how strategy usage differs between traces with prop match > 0.5 (high performance) and prop match < 0.5 (low performance).

Experimental setup. Three choices apply to both stages: which traces we analyse, how we summarize each one, and how we test for a difference. Our annotated traces are a stratified sample: per run, up to 60 traces with high performance and 60 with low performance, drawn from all solvable traces. The high-performance stratum reaches the full 60 per run on Eedi but is capped by availability on SciQ (37–44 per run; §A.5). Because we chose how many traces of each kind to annotate, high- and low-scoring traces appear here in near-equal numbers, which is not their ratio in the full data. Any measure that depends on that ratio is therefore distorted, including a correlation between strategy usage and the continuous proportional-match score. An average taken inside the high-scoring group, or inside the low-scoring group, does not depend on it. Both stages therefore compare the two groups directly rather than relating strategy usage to the continuous score, and neither needs reweighting.

We summarize each trace by its usage of the ten strategies in two ways: the absolute *count* of

Dataset	Model	Setting	Valid dropped	Invalid dropped
Eedi	DEEPSEEK-V3.2	reasoning	28% ($n=69$)	53% ($n=386$)
Eedi	DEEPSEEK-V3.2	CoT	10% ($n=4$)	64% ($n=56$)
Eedi	GLM-4.7	reasoning	17% ($n=34$)	34% ($n=118$)
Eedi	GLM-4.7	CoT	10% ($n=4$)	43% ($n=44$)
SciQ	DEEPSEEK-V3.2	reasoning	15% ($n=30$)	23% ($n=238$)
SciQ	DEEPSEEK-V3.2	CoT	8% ($n=7$)	9% ($n=30$)
SciQ	GLM-4.7	reasoning	6% ($n=20$)	17% ($n=228$)
SciQ	GLM-4.7	CoT	0% ($n=2$)	10% ($n=48$)

Table 28: Fraction of PLAUS/DISCR self-checks that express *negative* sentiment (arguing against keeping the candidate), split by whether the check’s target candidate is *valid* (equivalent to a human-authored distractor) or *invalid*. Inverse-probability-weighted to the population (§C.4); n is the raw count of checks per group. Checks on valid candidates are negative far less often than checks on invalid ones (the checks discriminate) but still at a non-trivial rate. CoT valid counts are small and are shown for completeness only.

each strategy, and its *share*, the count as a percentage of the number of tags in the trace; a difference between two shares is therefore in percentage points (pp). A count grows as a trace gets longer; a share does not. Every p -value below comes from a permutation test with 200,000 resamples, which needs no distributional assumptions: it reshuffles the high/low-scoring labels and asks how often chance alone reproduces a difference this large. DEEPSEEK-V3.2 and GLM-4.7 may use a given strategy at different rates overall, so we reshuffle labels only within a model; a difference between the two models can then never surface as a difference between high- and low-scoring traces. Both stages also control for problem difficulty, which we proxy by solution step count: the number of steps in a step-by-step reference solution generated for each Eedi problem, with the correct answer to the MCQ given as the required endpoint.

Stage 1: is the strategy profile different at all?

Before comparing individual strategies, we test per dataset and prompting setting whether the ten-strategy usage profile differs between high- and low-scoring traces at all, using PERMANOVA (Anderson, 2001; McArdle and Anderson, 2001) on the Bray–Curtis dissimilarity of the count profiles. Each trace is one vector of ten strategy counts, Bray–Curtis measures how far apart two such vectors are, and PERMANOVA asks whether dissimilarity between the high- and low-scoring groups exceeds dissimilarity within them, with significance obtained by reshuffling the high/low-scoring labels. It is an omnibus test: it evaluates the whole profile at once rather than one strategy at a time, so it pays no multiplicity penalty, and gating on it

means Stage 2 examines individual strategies only where there is evidence that something differs. On Eedi the profiles differ in both settings (reasoning $p = 0.008$, CoT $p = 0.032$; Table 29).

Problem difficulty is a confounder here: if low-scoring traces come from harder problems, and difficulty shapes which strategies are used, the profile difference would be a property of the problems rather than of the traces. We therefore restrict the permutation to within terciles of solution step count, exchanging labels only between traces from problems of comparable difficulty. Under that restriction the reasoning p is unchanged at 0.008 and the CoT p moves from 0.032 to 0.034, so the difference is not a difficulty artifact.

In contrast to Eedi, on SciQ no overall difference is detectable (reasoning $p = 0.16$, CoT $p = 0.06$); we therefore do not interpret per-strategy effects there.

Stage 2: which strategies drive the Eedi difference?

Both Eedi settings pass Stage 1, so we ask which strategies drive the difference there. For each strategy we want the high–low difference in usage. As discussed in Stage 1, two factors could confound it: we pool two models whose baseline usage differs, and low-scoring traces may come from harder problems. A permutation test on its own gives a p -value but no difference adjusted for either. We therefore use an analysis of covariance, the standard method for comparing two groups while holding a covariate fixed. It is a linear model predicting a strategy’s usage from three inputs: which model produced the trace, the problem’s step count, and whether the trace is high- or low-scoring. Fitting means choosing

one weight per input so that the predictions come as close as possible to the usage we observe. The weight on the high/low-scoring input is Δ : the high–low difference that remains once model identity and step count are held fixed. We report it twice, without step count (*naive*) and with it (*step-adj.*). Because each model contributes 60 high-scoring and 60 low-scoring traces, the naive Δ reduces exactly to $\text{mean}(\text{high}) - \text{mean}(\text{low})$. The linear model supplies the adjusted estimate only: every p -value below comes from a permutation test, so no distributional assumption is needed.

The p -value requires the same adjustment. Shuffling the high/low-scoring labels on their own would also break the association between usage and step count, and would therefore test a null hypothesis in which difficulty is randomized as well. A Freedman–Lane permutation (Freedman and Lane, 1983) avoids this: we refit the model without the high/low-scoring input, take the residuals (the usage left unexplained by model identity and step count), shuffle those, and refit. Only the high/low-scoring assignment is randomized, and the step-count structure is left in place. One further correction is needed because we test ten strategies per column: with ten independent tests the chance that at least one passes $p < 0.05$ by chance is about 40%, so we tighten the threshold with Benjamini–Hochberg (Benjamini and Hochberg, 1995) and report q -values instead: at $q < 0.05$, at most 5% of the differences we mark significant are expected to be false. All Δ values quoted below are step-adjusted.

One strategy stands out (Tables 30 and 31): error enumeration (ERR_DESC) is more frequent in low-scoring traces under both prompting settings, in count and in share. Under reasoning the count difference is significant (-15 occurrences, $q = 0.005$). The share difference is borderline significant in both settings (-2.9pp , $q = 0.048$ under reasoning; -4.2pp , $q = 0.044$ under CoT). Under CoT the count difference (-29 occurrences) is significant only before correction. Both models show the same sign for count and share, and step adjustment leaves the pattern unchanged, so problem difficulty does not explain the error-enumeration gap. Low-scoring traces also carry more tags overall, so a higher error-enumeration count alone could mean nothing more than more of everything; the share differences are what rule this out.

Two further share differences pass the correction: correct-answer reference (CORR, $+2.8\text{pp}$ un-

der CoT, $q = 0.044$) and task interpretation (INTER, $+2.6\text{pp}$ under reasoning, $q = 0.024$) take a larger share of high-scoring traces without a matching rise in count, and the CORR count is significantly lower under reasoning. High-scoring traces carry roughly 50 fewer tags than low-scoring ones in both settings (79 vs 128 per trace under reasoning, 23 vs 73 under CoT), and most of that gap is error enumeration and error simulation (ERR_SIM), plus outcome instantiation (INST) under reasoning. A count that is unchanged in a trace with fewer tags is a larger share, which is all the CORR and INTER gains are. The reverse pattern appears for ERR_SIM and INST: significant count gaps under reasoning that shrink or vanish as shares, so length accounts for those too. Read together, low-scoring traces are not drawn from harder problems, nor do they differ in which strategies appear; they carry more tags, and a larger part of those tags goes to enumerating student errors. Inflated error enumeration is the single behavioral signature that separates low- from high-scoring traces (§4.3).

Remaining confounders. Both stages control for problem difficulty via solution step count and for trace length by reporting shares alongside counts. Other properties of a problem go uncontrolled and could contribute to the contrast.

One is the size of a problem’s answer space. Some items admit very few candidate distractors. In the 37 problems of our annotated Eedi sample where two characters propose competing methods, the four options are always *Only Tom*, *Only Katie*, *Both* and *Neither*, so the three human-authored distractors are fixed in advance, a model recovers them easily, and proportional match saturates. Elsewhere the plausible answers are far more numerous. For $43.2 \div 10$ the human-authored distractors are 0.432, 33.2 and 43.02, each encoding a different misconception, and a model can reach further errors that are equally plausible but match none of the three.

Step count does not capture this. Those same 37 problems have the smallest answer space in the sample yet the longest reference solutions, averaging 4.95 steps against 3.44 for the other 290; the one asking which of two proposed next lines of $\frac{x}{3} - 5 = 12$ is correct takes eight steps to verify, while $43.2 \div 10$ takes two. Adjusting for step count therefore does not also adjust for answer-space size. If high- and low-scoring traces differ systematically in that size, and if it also shapes which strategies are used, part of the contrast would reflect problem

type rather than reasoning quality.

Two further factors are uncontrolled: the topic of a problem within a dataset, such as algebra against geometry, and whether its human-authored distractors encode recognizable misconceptions, which error simulation can recover, or are arbitrary alternatives. We therefore read Stage 2 as a description of how high- and low-scoring traces differ rather than as an isolation of a causal effect of strategy choice on performance.

Dataset	Setting	pseudo- F	p
Eedi	reasoning	3.95	0.008* (0.008)
Eedi	CoT	2.60	0.032* (0.034)
SciQ	reasoning	1.45	0.162
SciQ	CoT	2.10	0.058

Table 29: Stage 1 omnibus: PERMANOVA (pseudo- F on the Bray–Curtis dissimilarity of the ten-strategy count profile) for a difference between high-scoring (proportional match >0.5) and low-scoring (<0.5) traces, pooling both models with restricted permutation within model. For Eedi the parenthesised p is the difficulty-robust variant (permutation restricted within model $\times$ step-count tercile). * $p < 0.05$.

Where in the trace high- and low-scoring traces diverge. Stage 2 says which strategies differ between high- and low-scoring traces, but not where in a trace the difference falls. Figs. 9 and 10 show that, splitting the view of Fig. 2 by output quality for each model and prompting setting on Eedi. Each panel cuts a trace into five equal spans and gives the share of each strategy among the tags falling in one span, so it shows where a strategy is used rather than how much of it there is.

Under CoT the two groups differ most in the opening fifth. In high-scoring traces that span is almost all task interpretation (INTER) and correct-answer reference (CORR). In low-scoring traces error enumeration (ERR_DESC) is already prominent there, where it is near-absent in high-scoring ones. GLM-4.7 splits the same way on error simulation (ERR_SIM); DEEPSEEK-V3.2 does not. Under reasoning the shift appears in GLM-4.7 but not in DEEPSEEK-V3.2, whose two panels are near-identical across all ten strategies. The CoT panels for high-scoring traces rest on few tags, since those traces are short: fewer than 100 tags fall in the opening fifth for either model.

D Additional Experiments

D.1 Coverage of Strategies.

Beyond the characterization of LLMs’ distractor generation process in §4.2, it is also informative to examine not only how frequently each strategy occurs on average within a trace, but also how many traces contain at least one instance of that strategy.

Results. In Table 32, we observe that across both GLM-4.7 and DEEPSEEK-V3.2, reasoning traces show higher coverage of taxonomy strategies than CoT traces, especially for evaluative and iterative behaviors such as PLAUS, CURATE, and RECON. Core generative steps (e.g., CORR and INST) are nearly universal across models and prompting styles. On Eedi, GLM-4.7 in CoT shows less frequent usage of LINK but relatively higher usage of ERR_SIM than DEEPSEEK-V3.2, indicating minor differences between models.

Strategy	count		share (pp)	
	naive	step-adj.	naive	step-adj.
Task Interpretation	+0.45	+0.48	+2.52*	+2.57*
Correct Answer Ref.	-4.24*	-4.25*	+0.01	+0.02
Conceptual Link	-0.07	-0.07	-0.12	-0.12
Error Description	-15.35**	-15.42**	-2.84 [†]	-2.88*
Error Simulation	-11.87*	-11.94*	-1.56	-1.57
Outcome Instantiation	-15.36*	-15.45*	+0.37	+0.36
Plausibility Check	-1.57	-1.57	+0.26	+0.27
Discriminability Check	-0.18	-0.18	+0.39	+0.39
Final Set Curation	-0.13	-0.13	+0.57	+0.57
Reconsideration	-0.62	-0.62	+0.41	+0.40

Table 30: Stage 2: high–low Δ in per-trace strategy usage for Eedi / reasoning traces (positive = high-scoring traces use it more), pooling both models. *count*: absolute occurrences; *share*: percentage of the trace’s tags. *step-adj.*: adjusts for solution step count (Freedman–Lane). Permutation test, Benjamini–Hochberg across the ten strategies within each column (* $q < 0.05$, ** $q < 0.01$; [†] nominal $p < 0.05$). The step-adjusted Error Description share is borderline ($q = 0.048$).

Strategy	count		share (pp)	
	naive	step-adj.	naive	step-adj.
Task Interpretation	+0.06	+0.04	+1.12	+0.95
Correct Answer Ref.	-2.58	-2.93	+3.49**	+2.75*
Conceptual Link	+0.01	+0.01	+0.05	+0.06
Error Description	-27.43 [†]	-28.83 [†]	-4.52*	-4.15*
Error Simulation	-16.73	-17.65	-2.25	-1.71
Outcome Instantiation	-2.28	-2.16	+2.66	+2.53
Plausibility Check	-0.44	-0.42	-0.32	-0.37
Discriminability Check	-0.33	-0.31	+0.29	+0.35
Final Set Curation	-0.17	-0.13	-0.33	-0.23
Reconsideration	-0.27	-0.24	-0.18	-0.17

Table 31: Stage 2: high–low Δ in per-trace strategy usage for Eedi / CoT traces (positive = high-scoring traces use it more), pooling both models. *count*: absolute occurrences; *share*: percentage of the trace’s tags. *step-adj.*: adjusts for solution step count (Freedman–Lane). Permutation test, Benjamini–Hochberg across the ten strategies within each column (* $q < 0.05$, ** $q < 0.01$; [†] nominal $p < 0.05$). The step-adjusted Error Description and Correct Answer Ref. shares are borderline ($q = 0.044$ each).

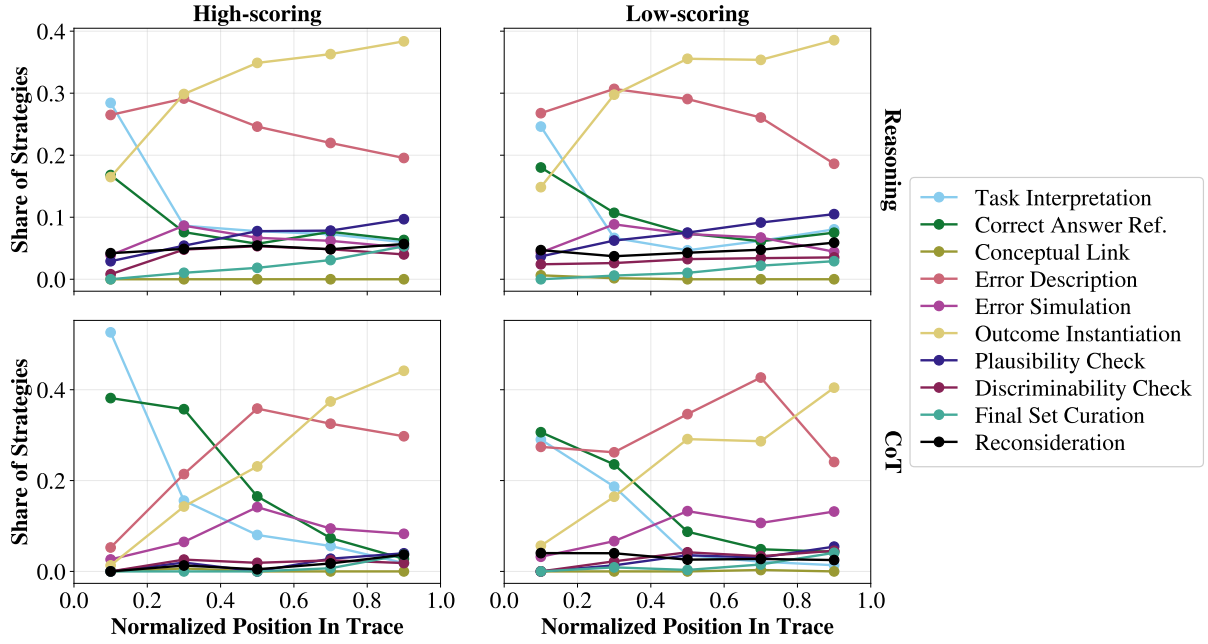


Figure 9: Strategy usage over normalized trace position in DEEPSEEK-V3.2 traces on Eedi, split by output quality (columns: high-scoring vs low-scoring, proportional match >0.5 vs <0.5) and prompting setting (rows). Within each fifth of the trace the strategy shares sum to one, as in Fig. 2.

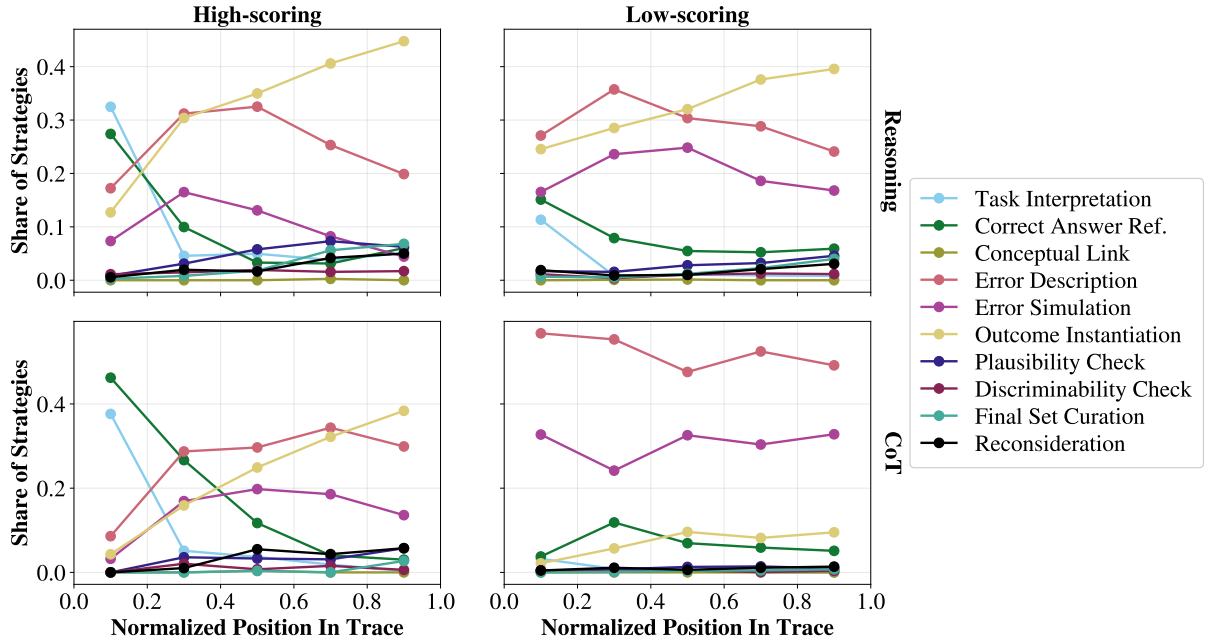


Figure 10: Strategy usage over normalized trace position in GLM-4.7 traces on Eedi, split by output quality and prompting setting, as in Fig. 9.

Model	Strategy	CoT		Reasoning	
		Eedi	SciQ	Eedi	SciQ
DEEPSEEK-V3.2	Task Interpretation	93.3	99.0	100.0	100.0
	Correct Answer Ref.	98.9	100.0	100.0	100.0
	Conceptual Link	1.1	94.8	1.7	99.0
	Error Description	96.7	27.8	96.7	31.0
	Error Simulation	43.9	0.0	55.6	0.0
	Outcome Instantiation	97.8	95.9	100.0	100.0
	Plausibility Check	65.0	84.5	96.7	91.0
	Discriminability Check	20.0	75.3	54.4	93.0
	Final Set Curation	57.2	85.6	93.9	100.0
	Reconsideration	31.1	11.3	97.2	63.0
GLM-4.7	Task Interpretation	65.0	100.0	98.3	100.0
	Correct Answer Ref.	100.0	100.0	100.0	100.0
	Conceptual Link	0.6	42.3	1.1	72.1
	Error Description	98.3	57.7	100.0	81.7
	Error Simulation	72.8	0.0	67.8	0.0
	Outcome Instantiation	98.9	98.1	100.0	100.0
	Plausibility Check	42.2	99.0	84.4	99.0
	Discriminability Check	9.4	88.5	32.2	97.1
	Final Set Curation	29.4	76.9	90.6	100.0
	Reconsideration	31.1	5.8	71.1	60.6

Table 32: Percentage of traces containing at least one occurrence of each taxonomy strategy (Table 1) for each model on both datasets.