LOOKALIKE: Consistent Distractor Generation in Math MCQs

Nisarg Parikh[1], **Alexander Scarlatos**[1*], **Nigel Fernandez**[1*],
Simon Woodhead[2], **Andrew Lan**[1]
University of Massachusetts Amherst[1], Eedi[2]
{nkparikh,nigel,ajscarlatos,andrewlan}@cs.umass.edu
simon.woodhead@eedi.co.uk

Abstract

Large language models (LLMs) are increasingly used to generate distractors for multiple-choice questions (MCQs), especially in domains like math education. However, existing approaches are limited in ensuring that the generated distractors are consistent with common student errors. We propose LOOKALIKE [3], a method that improves error–distractor consistency via preference optimization. Our two main innovations are: (a) mining synthetic preference pairs from model inconsistencies, and (b) alternating supervised fine-tuning (SFT) with Direct Preference Optimization (DPO) to stabilize training. Unlike prior work that relies on heuristics or manually annotated preference data, LOOKALIKE uses its own generation inconsistencies as dispreferred samples, thus enabling scalable and stable training. Evaluated on a real-world dataset of 1,400+ math MCQs, LOOKALIKE achieves 51.6% accuracy in distractor generation and 57.2% in error generation under LLM-as-a-judge evaluation, outperforming an existing state-of-the-art method (45.6% / 47.7%). These improvements highlight the effectiveness of preference-based regularization and inconsistency mining for generating consistent math MCQ distractors at scale.

1 Introduction

Multiple-choice questions (MCQs) are used in educational assessments (Nitko, 1996; Airasian, 2001; Kubiszyn and Borich, 2016) to evaluate student understanding across various subjects and grades (Thomas et al., 2025). An MCQ consists of a question stem and a set of options, including a correct answer and multiple incorrect alternatives, referred to as *distractors* (Fernandez et al., 2024; Feng et al., 2024). *Distractors* are incorrect answers that students reach by making an *error* while answering the question. It can be rooted in many ways, e.g., the student overgeneralizing to a new context, exhibiting an ingrained misconception, or simply slipping and being careless. Designing effective distractors can be crucial to the assessment and pedagogical aspects of MCQs (Simkin and Kuechler, 2005), since they help us identify student errors and prepare ways to mitigate them.

Hand-crafting high-quality distractors requires extensive human effort by content designers and teachers since it requires them to anticipate common student errors, which can be difficult in subjects like math. Therefore, recent works have leveraged artificial intelligence, especially large language models (LLMs), to automate this process. Previous works on *distractor* generation for MCQs have attempted to prompt LLMs to generate distractors (Feng et al., 2024), as well as fine-tune LLMs to generate possible student errors and then distractors caused by such errors, as shown in DiVERT (Fernandez et al., 2024). As noted in these works, the bottleneck in distractor generation performance is *consistency*: LLMs are often capable of identifying mathematically feasible errors, but struggle at following such erroneous instructions to arrive at the corresponding distractor (a similar finding was also made in (Sonkar et al., 2024a)). As shown in Table 1, both fine-tuned LLMs and the LLMs in DiVERT sometimes fail to follow the input error explanation to arrive at a consistent distractor. In the second example, the fine-tuned LLM fails to follow the error, "finds 13% of an amount rather than the percentage being asked", arriving at an inconsistent distractor (12) rather than the consistent distractor (5.2).

To address this limitation, one natural solution is to regularize an LLM-based distractor generator, which takes the question stem and an error as input, to enforce that the generated distractor matches the input error. To this end, we resort to preference optimization, specifically direct preference optimization (DPO) (Rafailov et al., 2023). DPO

[*] Equal Contribution.
[3] Code: https://github.com/umass-ml4ed/LookAlike

Proceedings of the 20th Workshop on Innovative Use of NLP for Building Educational Applications, pages 294–311

Question stem: Calculate: 130% of 40 = □	
Error	Distractor
Plausible error, plausible and consistent distractor.	
Added the values together instead of finding the percentage.	170
Plausible error, plausible but inconsistent distractor.	
Finds 13% of an amount rather than the percentage being asked.	12
Implausible error, plausible but inconsistent distractor.	
When solving a problem that requires an inverse operation (e.g. missing number problems), does the original operation.	90
Implausible error, implausible and inconsistent distractor.	
Does not understand that 100% is the whole amount.	20

Table 1: Examples of inconsistent error-distractor pairs generated by SFT (second and fourth pairs), and a state-of-the-art method, DiVERT (Fernandez et al., 2024) (third pair). LOOKALIKE mines generation inconsistencies for scalable preference optimization.

training requires *preference pairs* among outputs, i.e., a distractor that matches the input error and a distractor that does not. However, we empirically find two main challenges in using DPO to promote error-distractor consistency:

- Acquiring high-quality preference data typically requires costly manual annotation or unreliable synthetic heuristics (Li et al., 2023; Tan et al., 2024), which is difficult due to the nature of the distractor generation task.
- Models trained with DPO may deteriorate in quality after a few epochs (Pal et al., 2024; Liu et al., 2024b; Yan et al., 2025; Xu et al., 2024), showing training instability.

Contributions

In this paper, we introduce LOOKALIKE, proposing two methods to tackle these challenges and improve error-distractor consistency in math MCQs For the first challenge, we create preference pairs by generating *synthetic negative samples*: we evaluate LLM-generated errors, in addition to distractors, and use inconsistently generated errors and distractors as informative negative samples. This method creates meaningful signals that, when used in conjunction with consistent errors and distractors in DPO training, improve the consistency of LLMs in distractor generation. For the second challenge, we employ a regularization method in DPO training, which performs supervised finetuning (SFT) and DPO *alternatively* in consecutive training iterations, which performs better than combining them both into a single objective, as done in recent works (Liu et al., 2024b; Pal et al., 2024).

We conduct extensive experiments on a real-world dataset containing math MCQs used by hundreds of thousands of students, with human-written error descriptions behind each distractor. Results show that LOOKALIKE, compared to state-of-the-art baselines, improves distractor generation performance by up to 6%. We also show that LOOKALIKE improves error generation by up to 10%, using an LLM-as-a-Judge evaluation. We also provide qualitative examples and an error analysis highlighting the improved consistency of generated *errors* and *distractors*.

2 Background

In this section we formally introduce the tasks of *error* and *distractor* generation in math MCQs. We also detail a baseline for preference pair creation and a baseline for DPO regularization, combining preference alignment with supervised learning.

2.1 Task Definition

We consider an MCQ Q defined by its textual components: a **question stem** s, (optionally) its **correct answer** or **key** k, (optionally) an explanation of the key f, (optionally) question topic/concept tags t, and a set of **incorrect answer options** called ground truth distractors D. Each $d_i \in D$ is (optionally) associated with a corresponding ground truth human-written **error explanation** or error $e_i \in E$. All textual components above are represented as sequences of words and math symbols. We aim to model the space of plausible student errors E and their corresponding distractors D. We define two primary tasks:

1. **Error Generation:** Learn an LLM parameterized model, $LLM^{err}(s, k, f, t, d_i) \rightarrow \hat{e}_i$, that outputs an error description $\hat{e}_i$ consistent with the given input distractor d_i and MCQ.
2. **Distractor Generation:** Learn an LLM parameterized model, $LLM^{dis}(s, k, f, t, e_i) \rightarrow \hat{d}_i$, that outputs a distractor $\hat{d}_i$ consistent with the given error description e_i and MCQ.

2.2 Baseline: Preference Pairs from Ground-truth Error-Distractor Pairs

As a natural starting point, following a similar method from (Scarlatos et al., 2024b), one can

construct preference pairs for DPO as follows: For each question, there are multiple distractors ($D = d_1, d_2, \ldots, d_n$) and their corresponding errors ($E = e_1, e_2, \ldots, e_n$). As a baseline, for the error behind the i^{th} distractor, e_i, we can use d_i itself as the preferred response, and use the remaining distractors ($d_j \in D \setminus \{d_i\}$) as dispreferred responses. We use a similar procedure for the errors. We dub this method for preference pair construction as **DPO-GT** (ground truth). However, the number of dispreferred responses is limited by the number of human-written error-distractor pairs for the question. LOOKALIKE, on the other hand, creates preference pairs by generating *synthetic negative samples*, allowing for an arbitrary number of dispreferred responses for scalable preference optimization, resulting in improved consistency in both error and distractor generation (Section 3.1).

2.3 Baseline: DPO Regularization

Models trained with DPO have been shown to deteriorate in quality after a few epochs due to training instability (Pal et al., 2024; Liu et al., 2024b; Yan et al., 2025; Xu et al., 2024). Existing regularization techniques to improve DPO training stability include Regularized Preference Optimization (**RPO**) (Liu et al., 2024b), and DPO-Positive (**DPOP**) (Pal et al., 2024). RPO optimizes both the DPO loss and the SFT loss jointly, i.e., $L_{RPO} = L_{DPO} + \lambda\beta L_{SFT}$. The SFT loss uses the preferred response as the ground-truth completion. RPO suffers from conflicting gradient directions (Shi et al., 2023; Liu et al., 2024a), especially when the preference-based signal (DPO) incentivizes ranking decisions that are misaligned with the next-token prediction signal (SFT). DPOP uses the SFT objective as a penalty but their improvement is limited to preference pairs with high edit distances between them. LOOKALIKE, on the other hand, proposes an alternating optimization approach to stabilize DPO training, interleaving SFT and DPO training either at the per-batch or per-epoch level, resulting in improved consistency in both error and distractor generation compared to RPO and DPOP (Section 3.2).

3 Methodology

We now detail our framework, LOOKALIKE, which a) creates preference pairs by generating synthetic negative samples, and b) employs a DPO regularization technique of alternating optimization between SFT and DPO for better training stability, leading to improved error and distractor generation consistency.

3.1 Mining Preference Pairs via Inconsistencies for DPO

Prior work (Fernandez et al., 2024) has highlighted a significant issue of consistency in distractor generation performance, with LLMs struggling to follow error descriptions to arrive at corresponding distractors, examples of which are shown in Table 1. LOOKALIKE mines these generation inconsistencies as *synthetic negative samples* to create preference pairs for DPO training.

We visualize our preference pair creation in LOOKALIKE in Figure 1. For distractor generation, LLM^{dis} overgenerates a set of distractors for an input question stem and a ground-truth error. Each generated distractor is then compared against the ground-truth distractor. In our preference dataset, generated distractors that match the ground-truth distractor exactly are preferred responses, while those that do not exactly match the ground-truth distractor are dispreferred responses. A similar process is applied to create preference pairs for error generation, with exact string match[4] used to compare generated errors against the ground-truth error to form preference pairs.

Formally, given an MCQ dataset with samples, (s, e, d), where s is the question stem, e is the error description, and d is the corresponding distractor, we first train a distractor generation model, LLM^{dis}, to output the corresponding distractor through SFT. To create preference pairs, we then overgenerate multiple distractors $\hat{d} \in \hat{D}$ from the fine-tuned LLM^{dis} for each (s, e) pair. For each generated distractor $\hat{d}$, we check if $\hat{d}$ matches the ground-truth distractor d exactly. If yes, we add $\hat{d}$ as a preferred response, and if no, we add $\hat{d}$ as a dispreferred response in our distractor generation preference dataset. Having constructed the preference dataset, we further train our fine-tuned LLM^{dis} through DPO (Rafailov et al., 2023). A similar process is applied to form our error generation preference dataset which is then applied for DPO training of LLM^{err}. Creating preference pairs from the static ground-truth dataset is limited by the number of human-written annotations (Section 2.2). LOOKALIKE, on the other hand, uses generations from the currently fine-tuned LLM to

[4]LLM-as-a-Judge using GPT-4o-mini as a similarity measure led to lower performance.

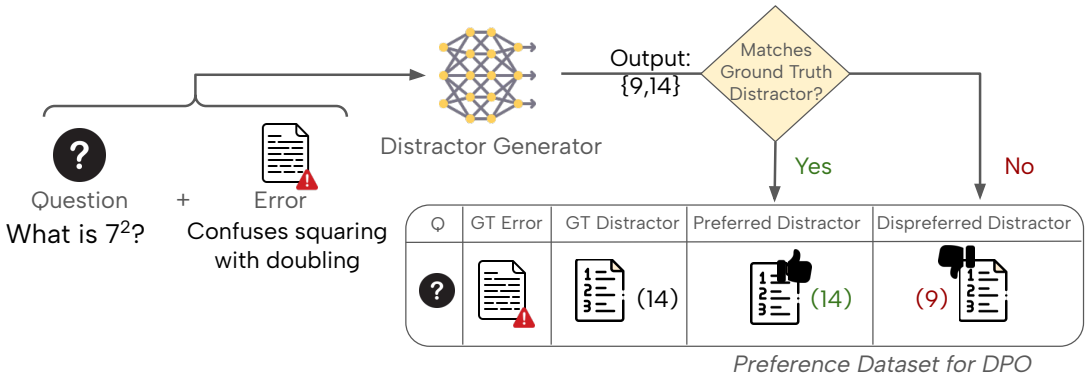

Figure 1: LOOKALIKE creates preference pairs by overgenerating a set of distractors for a question and error, and preferring those that match the ground-truth distractor exactly. An analogous process for error generation.

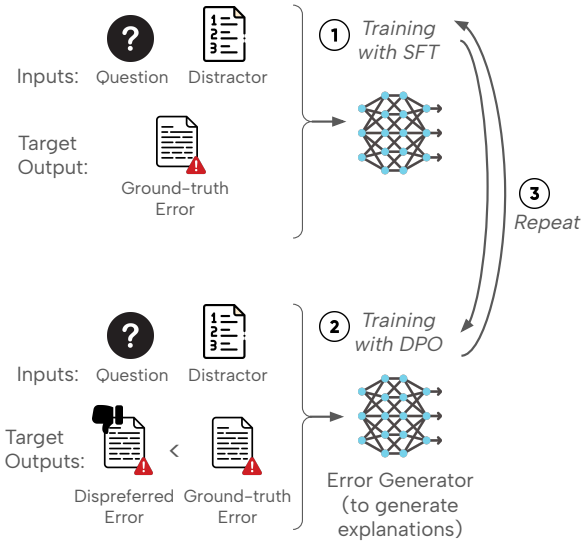

Figure 2: LOOKALIKE employs an alternating optimization strategy, switching between SFT and DPO objectives to regularize DPO training.

create an arbitrary number of dynamic preference pairs, with negative preference signals being more aligned with the inconsistency failure modes of the fine-tuned LLM.

3.2 DPO Regularization Through Alternating Optimization

We empirically observe that models trained with DPO deteriorate in quality after a few epochs due to training instability. We show examples of degradation in error generation quality over three training epochs in Table 2. We observe errors become more verbose with an increase in length and are out-of-distribution from the human-written errors as the number of DPO training epochs increases, as also shown in prior work (Park et al., 2024).

To mitigate this issue, we introduce a regularization strategy that trains the error/distractor-generation LLM by *alternating optimization*, i.e., by switching between SFT and DPO objectives during training, as shown in Figure 2. This alternating optimization allows the LLM to periodically recalibrate to the ground-truth distribution (via SFT) while remaining faithful to learning ranking preferences of consistent generations (via DPO). After each SFT optimization, the preference dataset is recomputed (Section 3.1) for the subsequent DPO optimization, using the currently trained LLM for better alignment, allowing for dynamic and scalable preference pair creation. We experiment with alternating between SFT and DPO optimization at two different levels: per-batch and per-epoch, picking the one giving better performance empirically. For both levels, the preference dataset is recomputed after every epoch.

Alternating Optimization Per-Batch. At each training step t, the LLM parameters θ are updated using a learning rate of η following:

$$\theta_{t+1} = \theta_t - \eta \nabla L(\theta_t), \tag{1}$$

where the loss function L alternates based on a batch-level schedule:

$$L(\theta_t) = \begin{cases} L_{SFT}(\theta_t), & \text{if batch } t \text{ is even} \\ L_{DPO}(\theta_t), & \text{otherwise} \end{cases} \tag{2}$$

Alternating Optimization Per-Epoch. As a coarser alternative, the loss function L alternates based on an epoch-level schedule:

$$L(\theta_t) = \begin{cases} L_{SFT}(\theta_t), & \text{if epoch } t \text{ is even} \\ L_{DPO}(\theta_t), & \text{otherwise} \end{cases} \tag{3}$$

Question	$\frac{3}{7}$ of a group of students are boys. What would be a possible ratio of boys to girls?
Key	$3:4$
Ground-truth Distractor	$3:10$
Ground-truth Error	Uses the denominator when converting from fractions to ratio, rather than numerator.
Generated Error (Epoch 1)	Includes the denominator when converting a fraction to a ratio.
Generated Error (Epoch 2)	When converting a fraction to a ratio, puts the other side of the ratio as the denominator.
Generated Error (Epoch 3)	When converting a fraction to a ratio, thinks you just use the numerator and denominator as the numbers in the ratio. Additionally, thinks you can use the denominator on its own as the total number of parts in a ratio.

Table 2: Error generation quality deteriorates over DPO training epochs without using regularization.

4 Experimental Evaluation

In this section, we detail our experiments on a real-world math MCQ dataset, evaluating the efficacy of LOOKALIKE in comparison with state-of-the-art baselines for both distractor generation and error generation.

4.1 Dataset

We conduct our experiments on a real-world math MCQ dataset from a large learning platform used by hundreds of thousands of students. The dataset consists of $1,434$ math MCQs, each containing a question stem, key, explanation of the key, topic/concept tags, and 3 distractors along with their respective teacher-written error descriptions explaining why a student might select that distractor. The MCQs are designed for students aged between 10 to 13 and span 41 distinct mathematical subtopics, including *Arithmetic*, *Fractions*, and *Solving Equations*. We split the dataset into training, validation, and test by questions to ensure no overlap across splits using a 72%-16%-12% proportion. See Appendix E for math MCQ examples.

4.2 Baselines

We compare LOOKALIKE with 3 baselines. The **SFT** baseline, used as a baseline in (Fernandez et al., 2024), fine-tunes an LLM to generate the corresponding distractor (or error) given the question and the error (or distractor) as input. The **DiVERT** (Fernandez et al., 2024) baseline employs a variational approach to learn an interpretable error space behind distractors. Post variational training, we use the error generation and distractor generation LLMs from DiVERT as baselines. We also compare against forming preference pairs from the ground-truth error-distractor pairs; we continue training the SFT baseline on this preference dataset using DPO and refer to the resulting model as **DPO-GT** (Section 2.2). For fairness, we regularize DPO training for DPO-GT by exploring all techniques (RPO, DPOP, our alternating per-batch optimization, and our alternating per-epoch optimization), and choose the regularization (per-epoch) that results in the best performance.

4.3 Metrics

Distractor Evaluation. Following prior work on distractor generation (Fernandez et al., 2024; Feng et al., 2024), we use **Exact match** as our evaluation metric to measure alignment between the generated distractor and the ground-truth distractor corresponding to a question and error.

Error Evaluation. Automated text similarity metrics like exact string match, ROUGE-L F1 (Lin, 2004), or BERTScore F1 (Zhang et al., 2020) are unsuitable for error evaluation given the open-ended and mathematical nature of errors. We therefore adopt an **LLM-as-Judge** (Liu et al., 2023; Zheng et al., 2023) evaluation, prompting GPT-4o-mini to evaluate if the generated error is mathematically equivalent to the ground-truth error given the question and corresponding distractor. We show our prompt in Appendix B.

4.4 Implementation Details

Following prior work (Fernandez et al., 2024), all methods use MetaMath-Mistral 7B (Yu et al., 2024b) as their base LLM, as we found it provides a suitable prior within the 7B parameter size models for mathematical reasoning. At test time, we use standard beam search with 10 beams for distractor generation, and diverse beam search (Vijayakumar et al., 2018) with 10 beams for error generation. Detailed hyperparameter settings for all methods are provided in Appendix A.

To ensure fair comparison, we limit LOOKALIKE 's synthetic generation to 3 distractors and 3

	Distractor Gen (Exact Match ↑)	Error Gen (LLM-as-Judge ↑)
SFT	44.76	46.68
DiVERT	45.64	47.72
DPO-GT	51.44	57.02
LOOKALIKE	**51.56**	**57.18**

Table 3: Cross-validation performance on distractor generation and error generation for all methods across 5 folds. LOOKALIKE outperforms SFT and the prior state-of-the-art method DiVERT (Fernandez et al., 2024), and is comparable to DPO-GT.

	Dis Gen (Exact M. ↑)	Error Gen (LLM-as-Judge ↑)
DPO-GT w/o Reg.	47.68	53.96
+ DPOP	47.80	52.74
+ RPO	49.14	52.44
+ Per-batch	49.66	55.74
+ Per-epoch	51.44	57.02
LOOKALIKE w/o Reg.	47.98	49.34
+ DPOP	49.38	49.44
+ RPO	49.60	49.66
+ Per-batch	50.84	**57.18**
+ Per-epoch	**51.56**	56.64

Table 4: Ablation study of various DPO regularization techniques. Our alternating (per-batch/epoch) optimization performs best for both DPO-GT and LOOKALIKE.

errors per training sample per epoch, resulting in a similar order of magnitude of training samples as DPO-GT. We also use the same training budget and regularization for both methods. All fine-tuned models, including SFT and DPO-based variants, were trained with LoRA to ensure parameter efficiency and consistency in comparison.

5 Results, Analysis and Discussion

In this section, we detail our experimental results. We quantitatively evaluate the quality of generated errors and distractors, qualitatively evaluate the consistency of generated errors through human evaluation, conduct an ablation study on DPO regularization techniques, and perform an error analysis on failed cases of error generation.

5.1 Quantitative Evaluation

Table 3 shows the average performance on distractor generation and error generation, across 5 cross-validation folds, for all methods. DPO-based methods, DPO-GT and LOOKALIKE, are trained using our alternating optimization technique for DPO regularization, choosing the alternating level (per-batch or per-epoch) that works best for downstream task performance. DPO-GT works best with per-epoch for both tasks, while LOOKALIKE works best with per-epoch for distractor generation, and per-batch for error generation.

Preference optimization using inconsistent error-distractor pairs improves consistency. LOOKALIKE outperforms SFT and the previous state-of-the-art baseline DiVERT (Fernandez et al., 2024), by a wide margin of 6.8% and 5.92% on distractor generation, and 10.5% and 9.46% on error generation performance, respectively. The improvement is statistically significant with p-values < 0.05 measured using a one-sample Wilcoxon signed-rank test (Rey and Neuhäuser, 2011). This result validates our idea of mining error-distractor inconsistencies as preference pairs for DPO training to improve both error and distractor generation consistency. Further, LOOKALIKE, although using synthetic negative samples drawn from its own inconsistent generations as preference pairs, is comparable in performance to DPO-GT, which uses human-written annotations as preference pairs, demonstrating the potential and flexibility of LOOKALIKE for scalable, domain-agnostic preference optimization.

Although the performance difference between LOOKALIKE and DPO-GT appears small (0.12% and 0.16% on distractor and error generation respectively), it is important to note that LOOKALIKE achieves this using automatically mined preference pairs from inconsistent generations, without relying on ground-truth labels, highlighting its scalability. Moreover, the improvement over DiVERT (5.9-10.5%) is substantial and statistically significant.

Alternating optimization is an effective DPO regularization. Table 4 shows an ablation study comparing different DPO regularization techniques to combat deterioration in generation quality (Pal et al., 2024) during DPO training. Existing approaches like DPOP (Pal et al., 2024) and RPO (Liu et al., 2024b) provide marginal gains up to 1.62% for distractor generation and 0.32% for error generation. Our alternation optimization, switching between SFT and DPO objective, at either the per-batch or per-epoch level, leads to the best performance for both, DPO-GT and LOOKALIKE, with performance gains up to 1.96% on the distractor generation task and 7.52% on the error generation task. These results show that alternating optimization effectively guides the LLM to periodically recalibrate to the ground-truth distribution (via SFT)

while remaining faithful to learning ranking preferences of consistent generations (via DPO).

5.2 Qualitative Case Studies

LOOKALIKE generates more consistent errors. Table 5 shows errors from LOOKALIKE compared to errors generated from SFT on two math questions. For the question on finding factors, SFT generates an overly generalized error applicable to many potential distractors, "Does not understand the term factor". On the other hand, LOOKALIKE generates a more specific error, "When asked for factors of an algebraic expression, thinks any part of a term will be a factor", consistent with the distractor. Similarly, for the question on simplifying algebraic terms, SFT generates an abstract error applicable to many distractors, "Tries to add or subtract unlike terms". On the other hand, LOOKALIKE generates a more specific and consistent error leading to the input distractor, "When collecting like terms, treats subtractions as if they are additions." We see similar patterns across other topics, with errors generated by LOOKALIKE being more specific and consistent with the input question and distractor. We also show qualitative examples of generated errors across all methods in Appendix D.

Error Analysis of LOOKALIKE. While LOOKALIKE outperforms SFT in generating more consistent errors and distractors, we observe some examples of generated errors that are inconsistent with the input question-distractor pair. One failure pattern observed is of *template overfitting*, where LOOKALIKE generates an error by overfitting to the error-distractor template of a similar question seen during training, generating errors that are consistent with other distractors from similar questions but not the input distractor. Table 8 in the Appendix shows two examples. We see that the generated error, "Has multiplied by the root power", is inconsistent with the input distractor 64, but upon inspection, is present as a ground-truth error and consistent with another question-distractor pair on the same topic.

5.3 Human Evaluation

Setup. We conduct a human evaluation on the quality and consistency of generated errors. We instruct two independent annotators with teaching experience to evaluate whether an error is consistent with a given input math question and corresponding distractor, choosing between a) yes, b) partially, and c) no. Our instructions to human annotators are provided in Appendix F.

We randomly select 40 math questions from our test set spanning a diverse range of topics. For each question, we include its ground-truth human-written error, the error generated by SFT, and the error generated by LOOKALIKE, for human evaluation. This process results in 120 errors, along with their corresponding questions and distractors, for human evaluation. We shuffle the 120 samples to avoid annotator bias.

Results. Table 6 shows the average of annotators' ordinal ratings on error explanations from the ground truth, SFT, and LOOKALIKE models. Ground truth errors scored the highest (mean = 0.812), followed by LOOKALIKE (0.587), and SFT (0.400). While LOOKALIKE does not match the human-authored ground truth, it significantly outperforms SFT on average, suggesting that preference-based regularization leads to more pedagogically consistent explanations.

We also measured agreement between annotators using quadratic-weighted Cohen's kappa, and found that error labels generated by LOOKALIKE led to the highest agreement (0.740), surpassing both SFT (0.659) and even the inter-annotator agreement on ground truth labels (0.415). This result suggests that errors generated by LOOKALIKE are easier for humans to interpret consistently, even if they are not always as plausible as ground truth explanations. We see a lower agreement on ground truth errors because their pedagogical nuance and potential generality made consistency judgments more subjective for annotators compared to the often more literal AI-generated errors.

Finally, we compared agreement between evaluations from human annotators to evaluations from GPT-4o-mini-based LLM-as-Judge, our reference metric for error generation. Agreement varied between annotators, with the first annotator showing moderate agreement (linear Kappa) with GPT-4o-mini (0.556 for LOOKALIKE-generated errors and 0.505 for SFT-generated errors), and the second annotator showing low agreement (0.314 for LOOKALIKE-generated errors and 0.409 for SFT-generated errors).

6 Related Work

Error-Distractor Generation for Math MCQs. Automated generation of math MCQs, and particularly their distractors, has progressed from

Topic	Finding factors	Simplifying terms
Question Stem	Which of the following is a factor of: $6n^2 - 9$?	Simplify the following expression by collecting like terms: $6x - 2y - x + 3y$.
Key	3	$5x + y$
Ground-truth Distractor	9	$7x + 5y$
Ground-truth Error	When asked for factors of an algebraic expression, thinks a term will be a factor.	When collecting like terms, treats subtractions as if they are additions.
SFT-Generated Error	Does not understand the term factor.	Tries to add or subtract unlike terms.
LOOKALIKE-Generated Error	When asked for factors of an algebraic expression, thinks any part of a term will be a factor.	When collecting like terms, treats subtractions as if they are additions.

Table 5: Examples showing errors generated from LOOKALIKE are more consistent than errors generated by SFT.

	Human	SFT	LOOKALIKE
Avg. Rating	0.812	0.400	0.587

Table 6: Average error consistency rating by human evaluators. LOOKALIKE generates more consistent errors than SFT.

template-based (rule-based and constraint-based) methods (Shin et al., 2019; Liang et al., 2018; Luo et al., 2024) to Large Language Model (LLM) approaches (Fernandez et al., 2024; Feng et al., 2024; Scarlatos et al., 2024a; Bitew et al., 2023; Chung et al., 2020). A critical challenge, however, remains the generation of high-quality distractors that accurately reflect common student errors and misconceptions (Alhazmi et al., 2024; Stasaski and Hearst, 2017). Current methods advance error representation using variational techniques (Fernandez et al., 2024), RAG-based methods (Yu et al., 2024a), and knowledge-bases (Ren and Q. Zhu, 2021).

Preference Optimization in Education. Preference learning techniques, including Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022) and its more stable, computationally efficient alternative Direct Preference Optimization (DPO) (Rafailov et al., 2023), are vital for aligning AI outputs with human judgments in education (Fahad Mon et al., 2023). Many recent approaches have used DPO (Lee et al., 2025; Sonkar et al., 2024b; Team et al., 2024; Ashok Kumar and Lan, 2024; Scarlatos et al., 2024b, 2025) but they do not handle some known failure modes of DPO related to inconsistent or out-of-distribution generation which the synthetic data generation of LOOKALIKE utilizes and the regularization of LOOKALIKE addresses. Other works mitigate these issues by providing regularization by using entropy (Shekhar et al., 2024), length-based rewards (Park et al., 2024), or the SFT objective (Liu et al., 2024b; Pal et al., 2024), LOOKALIKE improved on these by providing a simpler SFT-based regularization approach which requires less hyperparameter tuning and is easier to apply.

Challenges in Erroneous Instruction Following. Generating distractors from error descriptions, is an instance of the broader challenge of AI instruction following(Lou et al., 2024). AI systems, including LLMs, struggle with complex reasoning (Heo et al., 2024; Son et al., 2024), multi-step tasks (Chen et al., 2024; Wang and Lu, 2023; Fujisawa et al., 2024), and adhering to multiple constraints simultaneously (Wen et al., 2024), sometimes exhibiting a "curse of instructions" where performance degrades as complexity increases (Jang et al., 2022; Son et al., 2024). Generalization also poses a significant hurdle; models often fail to apply instructions to new tasks or in novel combinations (compositional generalization) (Cohen et al., 2025; Dan et al., 2021). These challenges can lead to inconsistencies where the generated output does not faithfully reflect the nuances of the input instruction (Jang et al., 2022; Son et al., 2024; Heo et al., 2024), a problem LOOKALIKE aims to mitigate in the context of error-distractor generation through targeted preference optimization.

7 Conclusion

In this paper, we introduced LOOKALIKE, a method that improves error-distractor consistency in math MCQs via preference optimization. LOOKALIKE uses two main innovations: a) mining synthetic preference pairs from model generation inconsistencies and b) alternating optimization by switching between SFT and DPO objectives to

stabilize training. Through extensive experiments on a real-world math MCQ dataset, we showed that LOOKALIKE outperforms the previous state-of-the-art method by a wide margin on both error generation and distractor generation. These improvements highlighted the potential of inconsistency mining and preference-based regularization for generating consistent math MCQ distractors at scale. We identify several limitations and avenues for future work. First, while LOOKALIKE improves error and distractor generation consistency, examples of inconsistent generations remain. Ideas for creating preference pairs using error generation and distractor generation models together could be a promising direction. Second, testing the generalizability of LOOKALIKE to math MCQs from unseen topics remains unexplored.

Limitations

While LOOKALIKE demonstrates improvements in generating consistent error-distractor pairs, it currently operates within the domain of middle-school mathematics. Extending the approach to other subjects like science or language arts may require minor modifications to the error and distractor representations.

Additionally, the current preference mining strategy relies on model-generated inconsistencies, which assumes the base model is sufficiently trained to surface pedagogically meaningful contrastive samples. In practice, we find that models pretrained on math data (e.g., MetaMath) meet this assumption, suggesting this is a broadly applicable approach rather than a bottleneck.

Our use of exact match to label non-matching outputs as dispreferred is conservative and intentionally strict; it helps emphasize high-confidence inconsistencies. Nonetheless, exploring softer similarity-based criteria or human judgments to refine preference mining is a valuable future direction.

Ethical Considerations

Our goal is to reduce educator workload by automating the generation of plausible distractors and their associated misconceptions, ultimately supporting teachers in providing more personalized feedback. However, we acknowledge a potential concern around over-reliance on AI-generated content in educational settings. While our system is designed to assist, not replace, educators, thoughtful deployment practices and educator-in-the-loop designs are encouraged.

The use of large language models (LLMs) introduces the standard risks of inherited biases or artifacts from pretraining data. In our case, these risks are minimal, as the domain of application (mathematical misconceptions) is highly constrained and less prone to sociolinguistic biases. Nevertheless, we encourage ongoing validation and periodic audits as best practices when deploying AI systems in learning environments.

Acknowledgments

This work is partially supported by Renaissance Philanthropy via the learning engineering virtual institute (LEVI) and NSF grants 2118706, 2237676, and 2341948. We thank Hasnain Heickal and Zhangqi Duan for helpful discussions and annotations regarding this work.

References

Peter Airasian. 2001. Classroom assessment: Concepts and applications. *McGraw-Hill, Ohio, USA*.

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.

Nischal Ashok Kumar and Andrew Lan. 2024. Improving socratic question generation using data augmentation and preference optimization. In *Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2024)*, pages 108–118, Mexico City, Mexico. Association for Computational Linguistics.

Semere Kiros Bitew, Johannes Deleu, Chris Develder, and Thomas Demeester. 2023. Distractor generation for multiple-choice questions with predictive prompting and large language models.

Xinyi Chen, Baohao Liao, Jirui Qi, Panagiotis Eustratiadis, Christof Monz, Arianna Bisazza, and Maarten de Rijke. 2024. The SIFo benchmark: Investigating the sequential instruction following ability of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*. Association for Computational Linguistics.

Ho-Lam Chung, Ying-Hong Chan, and Yao-Chung Fan. 2020. A BERT-based distractor generation scheme with multi-tasking and negative answer training strategies. In *Findings of the Association for Computational Linguistics: EMNLP 2020*. Association for Computational Linguistics.

Vanya Cohen, Geraud Nangue Tasse, Nakul Gopalan, Steven James, Matthew Gombolay, Ray Mooney, and Benjamin Rosman. 2025. Compositional instruction following with language models and reinforcement learning.

Soham Dan, Xinran Han, and Dan Roth. 2021. Compositional data and task augmentation for instruction following. In *Findings of the Association for Computational Linguistics: EMNLP 2021*. Association for Computational Linguistics.

Bisni Fahad Mon, Asma Wasfi, Mohammad Hayajneh, Ahmad Slim, and Najah Abu Ali. 2023. Reinforcement learning in education: A literature review. *Informatics*.

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 *Findings of the Association for Computational*

Linguistics: NAACL 2024. Association for Computational Linguistics.

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 *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Ippei Fujisawa, Sensho Nobe, Hiroki Seto, Rina Onda, Yoshiaki Uchida, Hiroki Ikoma, Pei-Chun Chien, and Ryota Kanai. 2024. Procbench: Benchmark for multi-step reasoning and following procedure.

Juyeon Heo, Miao Xiong, Christina Heinze-Deml, and Jaya Narain. 2024. Do LLMs estimate uncertainty well in instruction-following? In *Neurips Safe Generative AI Workshop 2024*.

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

Joel Jang, Seonghyeon Ye, and Minjoon Seo. 2022. Can large language models truly follow your instructions? In *NeurIPS ML Safety Workshop*.

Tom Kubiszyn and Gary Borich. 2016. Educational testing and measurement. *John Wiley and Sons, New Jersey, USA*.

Yooseop Lee, Suin Kim, and Yohan Jo. 2025. Generating plausible distractors for multiple-choice questions via student choice prediction.

Zhuoyan Li, Hangxiao Zhu, Zhuoran Lu, and Ming Yin. 2023. Synthetic data generation with large language models for text classification: Potential and limitations. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Chen Liang, Xiao Yang, Neisarg Dave, Drew Wham, Bart Pursel, and C. Lee Giles. 2018. Distractor generation for multiple choice questions using learning to rank. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*. Association for Computational Linguistics.

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, and Qiang Liu. 2024a. Conflict-averse gradient descent for multi-task learning.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Zhihan Liu, Miao Lu, Shenao Zhang, Boyi Liu, Hongyi Guo, Yingxiang Yang, Jose Blanchet, and Zhaoran Wang. 2024b. Provably mitigating overoptimization in RLHF: Your SFT loss is implicitly an adversarial regularizer. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Renze Lou, Kai Zhang, and Wenpeng Yin. 2024. Large language model instruction following: A survey of progresses and challenges.

Haohao Luo, Yang Deng, Ying Shen, See-Kiong Ng, and Tat-Seng Chua. 2024. Chain-of-exemplar: Enhancing distractor generation for multimodal educational question generation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics.

Anthony J. Nitko. 1996. Educational assessment of students. *Prentice-Hall, Iowa, USA*.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

Arka Pal, Deep Karkhanis, Samuel Dooley, Manley Roberts, Siddartha Naidu, and Colin White. 2024. Smaug: Fixing failure modes of preference optimisation with dpo-positive. *arXiv preprint arXiv:2402.13228*.

Ryan Park, Rafael Rafailov, Stefano Ermon, and Chelsea Finn. 2024. Disentangling length from quality in direct preference optimization.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Siyu Ren and Kenny Q. Zhu. 2021. Knowledge-driven distractor generation for cloze-style multiple choice questions. *Proceedings of the AAAI Conference on Artificial Intelligence*.

Denise Rey and Markus Neuhäuser. 2011. *Wilcoxon-Signed-Rank Test*. Springer Berlin Heidelberg.

Alexander Scarlatos, Wanyong Feng, Digory Smith, Simon Woodhead, and Andrew Lan. 2024a. 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, Mexico City, Mexico. Association for Computational Linguistics.

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

Alexander Scarlatos, Digory Smith, Simon Woodhead, and Andrew Lan. 2024b. Improving the validity of automatically generated feedback via reinforcement learning. In *Artificial Intelligence in Education*, pages 280–294, Cham. Springer Nature Switzerland.

Shivanshu Shekhar, Shreyas Singh, and Tong Zhang. 2024. See-dpo: Self entropy enhanced direct preference optimization.

Guangyuan Shi, Qimai Li, Wenlong Zhang, Jiaxin Chen, and Xiao-Ming Wu. 2023. Recon: Reducing conflicting gradients from the root for multi-task learning. In *The Eleventh International Conference on Learning Representations*.

Jinnie Shin, Qi Guo, and Mark J. Gierl. 2019. Multiple-choice item distractor development using topic modeling approaches. *Frontiers in Psychology*, Volume 10 - 2019.

Mark G Simkin and William L Kuechler. 2005. Multiple-choice tests and student understanding: What is the connection? *Decision Sciences Journal of Innovative Education*, 3(1):73–98.

Guijin Son, SangWon Baek, Sangdae Nam, Ilgyun Jeong, and Seungone Kim. 2024. Multi-task inference: Can large language models follow multiple instructions at once? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics.

Shashank Sonkar, Naiming Liu, MyCo Le, and Richard Baraniuk. 2024a. Malalgoqa: Pedagogical evaluation of counterfactual reasoning in large language models and implications for ai in education. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 15554–15567.

Shashank Sonkar, Kangqi Ni, Sapana Chaudhary, and Richard Baraniuk. 2024b. Pedagogical alignment of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*.

Katherine Stasaski and Marti A. Hearst. 2017. Multiple choice question generation utilizing an ontology. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics.

Zhen Tan, Dawei Li, Song Wang, Alimohammad Beigi, Bohan Jiang, Amrita Bhattacharjee, Mansooreh Karami, Jundong Li, Lu Cheng, and Huan Liu. 2024. Large language models for data annotation and synthesis: A survey. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

LearnLM Team, Abhinit Modi, Aditya Srikanth Veerubhotla, Aliya Rysbek, Andrea Huber, Brett Wiltshire, Brian Veprek, Daniel Gillick, Daniel Kasenberg, Derek Ahmed, Irina Jurenka, James Cohan, Jennifer She, Julia Wilkowski, Kaiz Alarakyia, Kevin R. McKee, Lisa Wang, Markus Kunesch, Mike Schaekermann, and 27 others. 2024. Learnlm: Improving gemini for learning.

Danielle R Thomas, Conrad Borchers, Sanjit Kakarla, Jionghao Lin, Shambhavi Bhushan, Boyuan Guo, Erin Gatz, and Kenneth R Koedinger. 2025. Does multiple choice have a future in the age of generative ai? a posttest-only rct. In *Proceedings of the 15th International Learning Analytics and Knowledge Conference*, pages 494–504.

Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R. Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. 2018. Diverse beam search: Decoding diverse solutions from neural sequence models. *Preprint*, arXiv:1610.02424.

Tianduo Wang and Wei Lu. 2023. Learning multi-step reasoning by solving arithmetic tasks.

Bosi Wen, Pei Ke, Xiaotao Gu, Lindong Wu, Hao Huang, Jinfeng Zhou, Wenchuang Li, Binxin Hu, Wendy Gao, Jiaxing Xu, Yiming Liu, Jie Tang, Hongning Wang, and Minlie Huang. 2024. Benchmarking complex instruction-following with multiple constraints composition. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics.

Shusheng Xu, Wei Fu, Jiaxuan Gao, Wenjie Ye, Weilin Liu, Zhiyu Mei, Guangju Wang, Chao Yu, and Yi Wu. 2024. Is dpo superior to ppo for llm alignment? a comprehensive study. In *Proceedings of the 41st International Conference on Machine Learning*.

Yuzi Yan, Yibo Miao, Jialian Li, YipinZhang, Jian Xie, Zhijie Deng, and Dong Yan. 2025. 3d-properties: Identifying challenges in DPO and charting a path

forward. In *The Thirteenth International Conference on Learning Representations*.

Han Cheng Yu, Yu An Shih, Kin Man Law, KaiYu Hsieh, Yu Chen Cheng, Hsin Chih Ho, Zih An Lin, Wen-Chuan Hsu, and Yao-Chung Fan. 2024a. Enhancing distractor generation for multiple-choice questions with retrieval augmented pretraining and knowledge graph integration. In *Findings of the Association for Computational Linguistics: ACL 2024*. Association for Computational Linguistics.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2024b. Metamath: Bootstrap your own mathematical questions for large language models.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

A Baselines and their Hyperparameters

We describe LOOKALIKE's baselines, as well as the hyperparameters used by LOOKALIKE and its baselines. We use MetaMath-Mistral 7B (Yu et al., 2024b) as our base LLM backbone for error and distractor generation across methods. For memory efficiency, we quantize the model weights into 8-bit integer representation and enable gradient checkpointing throughout training. Our implementation utilize the HuggingFace ecosystem, specifically the `transformers` (Wolf et al., 2020), `peft`, and `trl` libraries for finetuning. We perform training on NVIDIA L40 GPUs.

SFT. For the supervised finetuning (SFT) baseline we train the base model with Low-Rank Adaptation (LoRA) modules (Hu et al., 2022). LoRA is configured with a rank $r = 128$, $\alpha = 256$, and a dropout rate of 0.05. We perform SFT training for 5 epochs, with early stopping based on validation loss. We use the AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate of 2e-5. We use a batch size of 6.

DPO-based Baselines. For all DPO training, we set the hyperparameter $\beta = 0.5$ and the learning rate as 5e-6. We use a batch size of 6.

DPO-GT. As specified in 2.2 we have multiple errors and distractors associated with all questions, to create preference pairs for each pair of error and distractor, we place all the non-associated sample of either in the dispreferred pair while placing the specified samples in the preferred pairs.

RPO. For RPO (Section 2.3), we use $\lambda = 0.005$ as reported by them. We use the default implementations of RPO as provided in the trl library.

DPOP DPO-Positive (DPOP) (Pal et al., 2024) enhances DPO by preventing the model from merely reducing the likelihood of rejected examples where the edit distance in all pairs is large by using the SFT objective as a penalty. It introduces a constraint term to balance learning:

$$L_{DPOP} = L_{DPO} - \lambda \cdot \max(0, \log \frac{\pi_{ref}(y_w|x)}{\pi_\theta(y_w|x)}). \quad (4)$$

Here, we use $\lambda = 0.1$.

LOOKALIKE (Synthetic Data Generation). For the LOOKALIKE prefrence pairs (in Section 3.1) we generate 3 errors and distractors for each epoch of training to create negative preference samples, while considering the ground truth errors and distractors as the positive preference samples. We consider the top-k completions returned by beam search to get a set of $\hat{e}_i$ which augments the set of dispreferred responses further. We note that for all DPO training we use the SFT trained model as a warm start as with previous literature (Rafailov et al., 2023).

LOOKALIKE (Per-epoch and Per-batch Regularization). With the per-epoch and per-batch modes of LOOKALIKE (Section 3.2), we use the learning rate of 5e-6 for both DPO and SFT. For the per-epoch setting we perform one entire epoch of SFT after one epoch of DPO. Whereas for the per-batch setting if we run out of SFT batches while DPO training hasn't finished we rollback to the beginning of the SFT training data.

B LLM-as-a-judge

To assess whether two error explanations express the same underlying misconception, we use `GPT-4o-mini` as an automated judge. The model is provided with the question, distractor, and two error explanations, and asked to determine whether they are *mathematically equivalent* (Table 7), that is, whether they arise from the same conceptual

misunderstanding, regardless of wording. Below, we present an example of the prompt used in this evaluation.

This template was used for all pairwise comparisons of error explanations in the LLM-as-a-Judge evaluation.

C Error Analysis

While LOOKALIKE generally produces more specific and grounded error explanations, Table 8 also reveals some notable limitations. In the cube root example, the explanation "Has multiplied by the root power" reflects a plausible arithmetic confusion but doesn't clearly connect to the distractor value of 64, which results from cubing rather than misunderstanding cube roots. Similarly, in the number ordering case, the generated error implies digit-level misordering but lacks clarity on how this leads specifically to choosing "Only Katie." These examples suggest that while LOOKALIKE often captures fine-grained misconceptions, it can occasionally overgeneralize or introduce speculative reasoning not fully aligned with the distractor. This underscores the need for further refinement to ensure tighter alignment between the error explanation and the underlying choice.

D Comparing Errors across LOOKALIKE and its Baselines

Table 9 illustrates how different training methods produce qualitatively distinct reasoning errors across representative math questions. We observe a clear progression in the nature of these errors, reflecting the underlying supervision strategies. Models trained with SFT often generate surface-level mistakes indicative of limited conceptual understanding. In contrast, DiVERT tends to produce more structured but still incorrect procedural reasoning. Errors from DPO-GT reveal partial application of mathematical heuristics, suggesting more sophisticated—though still flawed—mental models. Finally, LOOKALIKE models (both per batch and per epoch) consistently produce errors that resemble common student misconceptions, such as overgeneralizing valid procedures or subtly misapplying familiar rules. This progression supports our claim that LOOKALIKE encourages more pedagogically meaningful error patterns, aligning closely with authentic human reasoning.

E Example MCQs from Real-world Math MCQ Dataset

We show example MCQs from the dataset in Table 10.

F Human Analysis Instructions

To evaluate the consistency of error explanations with corresponding distractor choices in multiple-choice math questions, we provided annotators with detailed guidelines, shown in Table 11. Annotators were instructed to examine each question item, which included a correct answer, a step-by-step solution, a distractor (incorrect answer), and an explanation for why a student might choose that distractor.

Annotators were asked to judge whether the explanation was:

- Yes: Clearly consistent with the distractor and plausibly explains the student error.
- Partially: Somewhat consistent, but vague, generic, or only loosely related to the distractor.
- No: Inconsistent or misleading; does not plausibly explain the choice of the distractor.

The instructions included concrete examples for each category to help calibrate judgment and ensure consistent annotation. These annotations were later used to analyze the quality of generated error explanations.

System Prompt.
You are a math education expert.
Given a question and a distractor (an incorrect student answer), determine whether two error descriptions are *mathematically equivalent*.

Definitions.

- An incorrect answer or distractor is a plausible but incorrect answer choice to the specified question.
- An error explanation or error is the misconception a student might make that leads them to choosing the specified distractor.
- Two error explanations are *mathematically equivalent* if they stem from the same core misunderstanding, regardless of wording.

Your response should include a brief justification (1–2 sentences) for whether the errors reflect the same or different misconceptions.
Always conclude with: "**Answer: Equivalent** or **Answer: Not Equivalent**".

Question and Metadata.
The question is: `<Question>`
The question topic is: `<Topic>`
The question concept is: `<Concept>`
The solution is: `<Solution from question to Correct Answer>`
The correct answer is: `<Correct Answer>`

Distractor (incorrect answer): `<Ground Truth Distractor>`

Error explanation 1: `<Ground Truth Error>`
Error explanation 2: `<Generated Error>`

Table 7: System prompt used to evaluate the mathematical equivalence of error explanations for a given distractor. The prompt positions the model as a math education expert tasked with identifying whether two misconceptions arise from the same underlying error.

Field	Cube Root	Indices, Powers and Roots
Question	$\sqrt[3]{8} = ?$	$3.52 + 2.75 =$
Distractor	64	5.27
Correct Answer	2	6.27
SFT Error Explanation	Divides by the order of the root.	Does not understand place value within a number.
LOOKALIKE Error Explanation	Has multiplied by the root power.	When adding decimals with a different number of decimal places, lines up the digits incorrectly.

Table 8: Comparison of error explanations for two different math topics. Examples show that LOOKALIKE also has some failure modes, discussed in greater depth in Section 5.2.

	Improper Fraction Conversion	**Gradient of a Line**
Question	Convert this into an improper fraction: $4\frac{2}{3}$	What is the gradient of this line? $5x + 3y = 15$
Correct Answer	$\frac{14}{3}$	$-\frac{5}{3}$
Ground-truth Distractor	$\frac{12}{3}$	$\frac{5}{3}$
Ground-truth Error	Forgetting to add the numerator to the whole part.	Applying the same operation instead of the inverse when solving.
SFT	Does not add the whole to the numerator when converting a mixed number to an improper fraction.	Believes a downward line has a positive gradient.
DiVERT	Thinks you add the number of wholes to the numerator when converting a mixed number to an improper fraction.	When solving an equation, uses the same operation rather than the inverse.
DPO-GT + Per batch	Does not include the whole amount when converting a mixed number to an improper fraction.	Believes the gradient of a line is given by the coefficient of x, even when the equation is not in the form $y = mx + c$.
LOOKALIKE + Per batch	Thinks you add the number of wholes to the numerator when converting a mixed number to an improper fraction.	When solving an equation, uses the same operation rather than the inverse.
DPO-GT + Per epoch	Thinks you can ignore the whole amount when converting a mixed number to an improper fraction.	When finding the gradient from the equation of a line in the form $ax + by = c$, believes b/a is the gradient.
LOOKALIKE + Per epoch	Thinks you add the number of wholes to the numerator when converting a mixed number to an improper fraction.	When finding the gradient from the equation of a line in the form $ax + by = c$, believes b/a is the gradient.

Table 9: Comparison of typical errors generated by each method for two representative math questions.

Question stem	Add brackets to this calculation to make the answer 7. $16 - 6 + 4 \div 2$
Topic	BIDMAS
Concept	Insert brackets to make a calculation correct
Solution	Inside the bracket we work left to right, so we get $14 \div 2$ which is 7.
Correct answer	$(16 - 6 + 4) \div 2$
Distractor 1	$16 - (6 + 4) \div 2$
Error 1	With order of operations brackets are done first, then division is done before subtraction. This would give us $16 - 10 \div 2 = 16 - 5 = 11$ NOT 7.
Distractor 2	$(16 - 6) + \frac{4}{2}$
Error 2	With order of operations brackets are done first, then division is done before subtraction. This would give us $10 + 4 \div 2 = 10 + 2 = 12$ NOT 7.
Distractor 3	$16 - 6 + (\frac{4}{2})$
Error 3	With order of operations brackets are done first, then division is done before subtraction. Putting the brackets around the division, will not change the order. $16 - 6 + (4 \div 2) = 16 - 6 + 2 = 12$ NOT 7.
Question stem	Which of the following answers gives the correct solutions to the quadratic expression below? $(x + 2)(x - 7) = 0$
Topic	Algebra
Concept	Solve quadratic equations using factorisation in the form (x + a)(x + b)
Solution	Setting each bracket equal to 0 we have x + 2 = 0 and x - 7 = 0. This tells us that x = -2 and x = 7.
Correct answer	$x = -2, x = 7$
Distractor 1	$x = 2, x = -7$
Error 1	Believes the solutions of a quadratic equation are the constants in the factorised form
Distractor 2	$x = 2, x = 7$
Error 2	Believes the solutions of a quadratic equation are the absolute values of the constants in the factorised form
Distractor 3	$x = -2, x = -7$
Error 3	Believes the solutions of a quadratic equation are the negative of the absolute values of the constants in the factorised form

Table 10: Example MCQs from the real-world math MCQ dataset.

In this task, you'll evaluate error explanations for student errors in math multiple-choice questions. For each item, you'll see:

1. The **question**
2. The **correct answer** choice
3. A **solution** which shows how a student can reach the correct answer choice
4. A **distractor** (an incorrect answer choice)
5. An **error** explanation describing why a student might choose the distractor

Your Task
Annotate if each error explanation is consistent with the distractor (mark Yes), is generic, vague, or partially consistent (mark Partially) or has nothing to do with the distractor, or is misleading (mark No).
Use your best judgment when assigning ratings. Some examples are:

Example 1 (Marking **Yes**):
Question: Add brackets to this calculation to make the answer 7. $16 - 6 + 4 \div 2$

Correct Answer:$(16 - 6 + 4) \div 2$
Solution: Inside the bracket we work left to right, so we get 14 ÷ 2 which is 7.
Distractor: $16 - (6 + 4) \div 2$
Error: Carries out operations from left to right regardless of priority order.
Mark **Yes**

Example 2 (Marking **Partially**):
Question: $\frac{3}{7}$ of a group of students are boys. What would be a possible ratio of boys to girls?

Correct Answer: $3 : 4$
Solution: For every 7 students, 3 are boys and 4 are girls. The ratio is then 3:4.
Distractor: $3 : 7$
Error: Uses the denominator when converting from fractions to ratio, rather than numerator.
Mark **Partially**

Example 3 (Marking **No**):
Question:When $h = 5$ $h^2 =$

Correct Answer: 25
Solution: If $h = 5$, $h^2 = h \times h = 5 \times 5 = 25$.
Distractor: 7
Error: Multiplies by the index.
Mark **No**

Table 11: Instructions provided to human annotators used to evaluate the consistency of error explanations for a given distractor.