

# DiVERT: Distractor Generation with Variational Errors Represented as Text for Math Multiple-choice Questions

Nigel Fernandez<sup>1\*</sup>, Alexander Scarlatos<sup>1\*</sup>, Wanyong Feng<sup>1</sup>  
Simon Woodhead<sup>2</sup>, Andrew Lan<sup>1</sup>

<sup>1</sup>University of Massachusetts Amherst, <sup>2</sup>Eedi

{nigel, ajscarlatos, wanyongfeng, andrewlan}@cs.umass.edu

simon.woodhead@eedi.co.uk

## Abstract

High-quality distractors are crucial to both the assessment and pedagogical value of multiple-choice questions (MCQs), where manually crafting ones that anticipate knowledge deficiencies or misconceptions among real students is difficult. Meanwhile, automated distractor generation, even with the help of large language models (LLMs), remains challenging for subjects like math. It is crucial to not only identify plausible distractors but also understand the *error* behind them. In this paper, we introduce DiVERT (**D**istractor Generation with **V**ariational **E**rrors **R**epresented as **T**ext), a novel variational approach that learns an interpretable representation of errors behind distractors in math MCQs. Through experiments on a real-world math MCQ dataset with 1,434 questions used by hundreds of thousands of students, we show that DiVERT, despite using a base open-source LLM with 7B parameters, outperforms state-of-the-art approaches using GPT-4o on downstream distractor generation. We also conduct a human evaluation with math educators and find that DiVERT leads to error labels that are of comparable quality to human-authored ones.

## 1 Introduction

Multiple-choice questions (MCQs) are arguably the most common form of questions found in standardized tests. Each MCQ contains a question *stem* that states the context of the questions and the task to be completed, and a series of *options*: a *key*, i.e., the correct answer, embedded among several other *distractors*, i.e., incorrect answers. See Figure 1 for an example. MCQs are widely used in real-world, large-scale educational/psychological tests and surveys, mainly due to the ease in automated grading (Nitko, 1996; Airasian, 2001; Kubiszyn and Borich, 2016). However, constructing a good set of distractors can be quite challenging: On the

one hand, they should correspond to clear errors in the factual recall or reasoning process required by the question’s task. On the other hand, these errors should not be too obvious such that no student would select the distractors. Therefore, designing high-quality MCQs that measure specific knowledge components/concepts/skills and more importantly, corresponding distractors good at capturing specific knowledge deficiencies among real students/test takers is very important to the development of high-quality MCQs.

The typical approach to MCQ distractor development primarily relies on extensive human effort, which can be burdensome for educators, which motivated the development of *automated* approaches. Prior work on automated distractor generation primarily focuses on i) cloze tasks in language learning to assess vocabulary recall or grammatical knowledge and ii) reading comprehension question answering to assess comprehension of a given text or article (Alhazmi et al., 2024). Approaches include using knowledge graphs, encoder-decoder models, and with help from large language models (LLMs). See Section 7 for a more detailed review of related work.

For other subjects where common tasks require (possibly complex) reasoning ability, such as math, there exist relatively few approaches to automated MCQ distractor generation. In these domains, generating high-quality distractors is more challenging than in language learning or reading comprehension: distractors need to reflect abstract mathematical misconceptions and/or procedural errors in the mathematical reasoning process. Earlier works either use symbolic, rule-based approaches to generate distractors (Tomás and Leal, 2013; Prakash et al., 2023), which have limited generalizability beyond template-based questions, or sample incorrect generations during a math problem solving process as distractors (Dave et al., 2021). More recently, (Feng et al., 2024; Scarlatos et al., 2024a)

---

\*These authors contributed equally to this work.

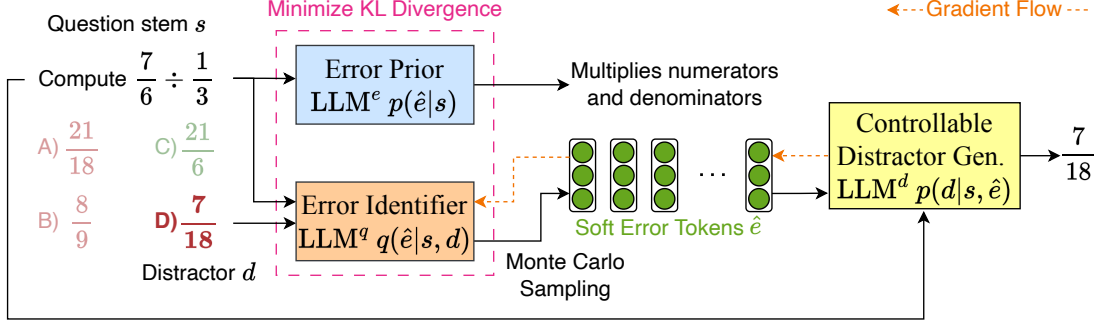


Figure 1: Overview of DiVERT’s variational pipeline for error explanation and distractor generation in math MCQs.

investigate a wide array of LLM-based approaches for math MCQ generation, particularly focusing on the plausibility of distractors, i.e., how likely is a distractor going to be selected among real students.

However, to the best of our knowledge, no existing approach has attempted to generate *explanations* of errors underlying MCQ distractors. In assessment scenarios where one’s goal is to measure the overall ability of a student/test taker, being able to generate a good set of distractors may be sufficient. However, in real-world educational scenarios where one’s goal is to maximize the learning gain of students, being able to *interpret* the cause of error behind a distractor is highly important; this information can be used for student knowledge *diagnosis*, i.e., identifying areas where they lack sufficient knowledge (VanLehn, 1982) or even pinpoint specific misconceptions they exhibit (Wang et al., 2021). Such diagnosis can be used to provide *feedback* to both teachers, to help them better understand students’ learning progress and directly to students in real-time, through intelligent tutoring systems (Ritter et al., 2007), online learning platforms (Heffernan and Heffernan, 2014), or via chatbots powered by LLMs (Academy, 2024). Therefore, interpreting the errors behind students selecting specific distractors in math MCQs is important yet challenging, partly due to the mathematical reasoning processes required by these questions.

### 1.1 Contributions

1. We introduce DiVERT<sup>1</sup> (**D**istractor Generation with **V**ariational **E**rrors **R**epresented as **T**ext), a novel variational approach to jointly learn error representations in math MCQs as well as generate distractors corresponding to errors.

2. We add interpretability to DiVERT by using LLMs to parameterize all distributions in our variational approach. This design enables us to represent and interpret errors as text tokens.
3. We conduct extensive quantitative and qualitative experiments on a real-world math MCQ dataset. We find that DiVERT, by jointly learning to represent errors and generate distractors, outperforms state-of-the-art approaches on distractor generation.
4. We conduct a human evaluation with math educators and find that DiVERT leads to error explanations comparable in quality to human-authored ones, and significantly outperforms GPT-4-generated errors despite using a much smaller base LLM with 7B parameters.

## 2 Problem Formulation

We denote each math MCQ as  $Q = \{s, k, f, t, D, E\}$ , which contains a set of textual components, including the question stem  $s$ , the key  $k$ , (optionally) an explanation of the key  $f$ , (optionally) question topic/concept tags  $t$ , and a set of distractors  $D$ : we denote each distractor itself as  $d_i \in D$ , with  $e_i \in E$  denoting the *error explanation* associated with the distractor, e.g., a misconception. In this paper, we do not consider MCQs with figures or diagrams since the open-source LLMs we work with cannot process visual input. All of these textual components are sequences of words and math symbols and we denote them as a series of tokens  $\{w_1, \dots, w_L\}$ , with  $L$  being the length of the sequence.

We study the task of learning an interpretable space of errors behind distractors in math MCQs. Since it is difficult to quantitatively evaluate error representations, we also study the downstream task of distractor generation (Feng et al., 2024). Specifi-

<sup>1</sup>Code: <https://github.com/umass-ml4ed/divert>

cally, our two major goals are:

1. **Error generation:** From a set of real-world math MCQs, learn an error representation space underlying distractors. Possible errors include insufficient knowledge on required skills or exhibiting specific misconceptions.
2. **Distractor generation:** For each plausible error, generate corresponding distractor(s), i.e., incorrect answer(s) that an incorrect approach with that error leads to.

We select distractor generation as the downstream task since performance on this task is easily quantifiable; however, there are other meaningful tasks, such as feedback message and tutoring dialogue turn generation, where current methods struggle to implicitly identify student errors (McNichols et al., 2024; Scarlatos et al., 2024b), yet knowing what error a student made significantly improves LLMs’ performance on these tasks (Jurenka et al., 2024). We leave exploring such tasks for future work.

We formulate the error generation task as learning a function that outputs a plausible error in an MCQ, given the question stem and attributes including the key and (optionally) explanation and topic tags, i.e.,  $g^{\text{err}}(s, k, f, t) \rightarrow \hat{e}$ . Similarly, we formulate the distractor generation task as learning a function that outputs a distractor that corresponds to a plausible error, i.e.,  $g^{\text{dis}}(s, k, f, t, \hat{e}) \rightarrow \hat{d}$ .

### 3 Methodology

We now detail our approach to error representation learning and distractor generation in math MCQs.

#### 3.1 Variational Approach

There can be numerous plausible errors among real students for each math MCQ, all of which may lead to different distractors. For example, in the MCQ in Figure 1 on fraction division, the distractor  $\frac{7}{18}$  may result from a student exhibiting the error “multiplies numerators and denominators”, while the distractor  $\frac{8}{9}$  may result from the error “adds numerators and denominators”. Without explicit error annotations, an error  $e$  is a latent variable underlying a distractor  $d$ . Thus, the observed likelihood of a distractor given a question stem  $s$  is

$$p(d|s) = \sum_{e \in \mathcal{E}} p(e|s)p(d|s, e), \quad (1)$$

where  $p(e|s)$  is the probability that the error  $e$  can be made on question  $s$ ,  $p(d|s, e)$  is the probability that  $d$  corresponds to the error  $e$ , and  $\mathcal{E}$  is the

space of all possible errors, which can be computationally intractable depending on how the errors are represented. While there are many potential representations of the errors, such as discrete categories or continuous latent vectors, we choose to represent each error as a *sequence of textual tokens*, so that the learned underlying representations of distractors are *interpretable* by both humans and language models.

Given this perspective, we naturally use a variational approach to learn an approximation of the large latent error space. Similar to variational auto-encoders (Kingma and Welling, 2014), we jointly learn the latent distribution and maximize the likelihood of the data by maximizing the evidence lower bound (ELBO) on the observed data log-likelihood, given by

$$\begin{aligned} \log p_\theta(d|s) &\geq \text{ELBO}(d|s) = \mathcal{L}(\theta, \phi) \\ &= \mathbb{E}_{q_\phi(e|s, d)} [\log p_{\theta_d}(d|s, e)] \\ &\quad - \beta D_{\text{KL}}(q_\phi(e|s, d) \parallel p_{\theta_e}(e|s)), \end{aligned} \quad (2)$$

where  $\theta_e$ ,  $\theta_d$ , and  $\phi$  are the parameters of the prior model, distractor likelihood model, and variational model, respectively. The parameter  $\beta > 0$  controls the balance between the distractor reconstruction loss and the KL divergence between the approximate posterior and the prior.

To maximize the objective above, we jointly train three models, each parameterized by an LLM:

1. The **error prior model**  $p_{\theta_e}(e|s)$ : This model generates a textual explanation of an error that students can make given the MCQ’s stem.
2. The **controllable distractor generation model**  $p_{\theta_d}(d|s, e)$ : This model generates a distractor that is the resulting incorrect answer after making a specific error in this question.
3. An **error identifier** model  $q_\phi(e|s, d)$ : This approximate posterior model generates the error behind a question-distractor pair, and acts as the variational distribution during training.

We note that there can be many choices for the variational distribution  $q(\cdot)$ ; we choose  $q(e|s, d)$  since it can be useful in practice by allowing us to recover textual errors from distractors. We leave experimenting with other useful approximate distributions such as  $q(e|t)$  or even simply  $q(e)$  for future work. Next, we detail how we train all three models in a connected fashion, with the pipeline shown in Figure 1.

### 3.2 DiVERT

As discussed above, to promote interpretability in error representations, we parameterize all distributions in our variational approach using LLMs. We use a different LLM for each distribution; each is fine-tuned from the same base LLM using QLoRA (Dettmers et al., 2024). We denote them as  $\text{LLM}^e$ ,  $\text{LLM}^d$ , and  $\text{LLM}^q$ . Next, we detail a series of novel methods that we introduce to enable DiVERT to effectively perform the tasks of error learning and distractor generation.

**Differentiable Learning through Discrete Tokens.** The ELBO in Equation 2 can easily be approximated by Monte Carlo simulation, with samples drawn from  $q_\phi(e|s, d)$  (see Supplementary Material A). However, since we parameterize  $q_\phi$  with an LLM, these samples are sequences of discrete text tokens. Therefore, we cannot simply use the reparameterization trick to sample from  $q_\phi$ . Using a similar approach to (Liu et al., 2023), we use soft tokens, i.e., differentiable versions of hard, discrete text tokens during training to enable the flow of gradients. Specifically, for the  $k$ -th error token  $\hat{e}_k$  generated by  $\text{LLM}^q$ , we replace its discrete embedding  $\mathbf{v}_{\hat{e}_k}$  with  $\sum_j p_{k,j} \mathbf{v}_j$ , where  $p_{k,j}$  is the probability of generating token  $j$  at position  $k$  from  $\text{LLM}^q$  and  $\mathbf{v}_j$  is the embedding vector of token  $j$  in the vocabulary of  $\text{LLM}^q$ . This approximation enables us to sample discrete tokens while differentiating through continuous vectors during training. We calculate  $p_{k,j} = \text{softmax}(\mathbf{z}_k / \lambda)_j$ , where  $\mathbf{z}_k$  is the output logit vector from  $\text{LLM}^q$  at position  $k$  and  $\lambda > 0$  is a temperature parameter. Following prior work (Jang et al., 2017), we initialize  $\lambda$  to 1 and exponentially anneal it to 0.1 during training. This process means that the soft tokens are smooth when training starts to enable better gradient flow, while later becoming closer to hard tokens.

**Initialization with Supervised Fine-tuning.** Despite LLMs exhibiting better and better mathematical reasoning capabilities, their capacity in understanding errors (Sonkar and Baraniuk, 2023) and generating distractors that correspond to errors (Feng et al., 2024) remains surprisingly poor. Therefore, we solicit a collection of error labels behind question-distractor pairs,  $(s, d)$ , from math educators. Then, we fine-tune all three LLMs in their respective formats using this annotation data to initialize DiVERT’s three LLM components. Without this step, base LLMs, even ones that perform

well on question answering tasks, generate low-quality error explanations that hurt performance, especially during early training stages. In Section 5, we demonstrate that using a small portion of error labels for warm-up fine-tuning helps DiVERT’s ability to learn from unlabeled  $(s, d)$  pairs.

**Q Regularization.** Since the space of plausible errors can be large, we add a regularization term in our loss function to prevent the approximate posterior distribution  $q_\phi(e|s, d)$  from deviating too much compared to its initialized version after supervised fine-tuning,  $q_{\phi_{\text{init}}}$ . This term, weighted by a balancing parameter  $\alpha > 0$ , controls the amount of exploration in our error tokens sampled from  $q_\phi(e|s, d)$  and prevents it from diverging during training. Concretely, the regularization is given by

$$\mathcal{L}_{\text{reg}} = \sum_{k=1}^{|\hat{e}|} D_{\text{KL}}(q_{\phi_{\text{init}}}(\hat{e}_k|s, d) || q_\phi(\hat{e}_k|s, d)), \quad (3)$$

with our final training objective to minimize becoming

$$-\mathcal{L}(\theta, \phi) + \alpha \mathcal{L}_{\text{reg}}. \quad (4)$$

We note that this term is similar to the KL penalty used when training LLMs with reinforcement learning (Ouyang et al., 2022), specifically NLPO (Ramamurthy et al., 2022) that also uses a token-level penalty. However, the key difference is that we directly backpropagate gradients from this loss rather than use it to form a reward.

**Overgenerate-and-rank for Errors and Distractors.** At test time, we first generate a set of  $N_e$  errors, denoted as  $\hat{E}$ , through  $\text{LLM}^e$ , using diverse beam search (Vijayakumar et al., 2018) for decoding, to promote diversity among generated errors. Then, for each generated error  $\hat{e} \in \hat{E}$ , we generate  $N_d$  distractors through  $\text{LLM}^d$  using standard beam search since the distractors exhibit much less variation under a specific error. Finally, we rank all  $N_e \times N_d$  candidate distractors in  $\hat{D}$  by their associated beam scores and select the top- $K$  (Ashok Kumar et al., 2023).

## 4 Experimental Evaluation

In this section, we detail our experiments on a real-world math MCQ dataset. For quantitative evaluation, we compare DiVERT against state-of-the-art approaches and strong baselines on the task of distractor generation. For qualitative evaluation, we perform a human evaluation of the generated error explanations with math educators.



## 4.1 Dataset Details

We work with a real-world math MCQ dataset from the Eedi learning platform<sup>2</sup>, containing 1,434 MCQs written in English, each with a set of 3 distractors. We collect error labels from middle school math teachers for each question-distractor pair, explaining why a student may select that distractor. The questions are designed primarily to assess students aged between 10 to 13, on 41 unique subtopics, including “Basic Arithmetic”, “Fractions”, and “Solving Equations”. We divide the dataset into train-val-test splits by *questions* to ensure no overlap in the MCQ stem across the splits, resulting in roughly a 72%-16%-12% split over question-distractor pairs. See Supplementary Material D for statistics, and Supplementary Material G for MCQ examples.

## 4.2 Metrics

**Error Evaluation.** The open-ended and mathematical nature of errors makes automated text similarity metrics like ROUGE-L F1 (Lin, 2004) and BERTScore F1 (Zhang et al., 2020) unsuitable. Therefore, we conduct a **human evaluation** of generated errors, which we detail in Section 6. For completeness, we report error evaluation on automated metrics in Supplementary Material E.

**Distractor Evaluation.** Following prior work on automated distractor generation (Feng et al., 2024), we use alignment-based metrics to measure how well the  $K$  generated distractors align with ground-truth human-authored ones. The first metric, **Exact match** ( $h_e$ ), measures whether all three human-authored distractors in  $D$  are matched exactly by some subset of the generated distractors  $\hat{D}$ . Similarly, **Partial match** ( $h_p$ ) measures whether at least one generated distractor matches human-authored ones. Concretely, these binary metrics are defined as  $h_e(D, \hat{D}) = 1$  if  $D \subseteq \hat{D}$ , and 0 otherwise, while  $h_p(D, \hat{D}) = 1$  if  $\hat{D} \cap D \neq \emptyset$ , and 0 otherwise. The third, continuous metric, **Proportional match** ( $h_n$ ), measures the portion of human-authored distractors that match generated ones, defined as  $h_n(D, \hat{D}) = |\hat{D} \cap D|/3$ . We compute all metrics averaged across all MCQs in the test set and report percentages. We vary  $K \in \{3, 10\}$ , similar to the setup for metrics such as MAP@K. The Proportional match metric is most important since it is more robust than the other two.

## 4.3 Baselines

We compare our variational approach, DiVERT, to state-of-the-art distractor generation approaches as well as several strong baselines, outlined below.

**Prompting Baselines.** We compare DiVERT to two prompting-based approaches proposed in (Feng et al., 2024) using the state-of-the-art LLM GPT-4o (OpenAI, 2024a). In contrast to our approach, these approaches generate all distractors in one pass, without considering errors behind them. The first is **kNN**, their best-performing approach, where we use the 3 most similar MCQs in the training set as in-context examples and teacher-written errors as chain-of-thought reasoning to aid distractor generation. We only use questions that have errors for all distractors as examples. The second is **CoT**, where no in-context examples are given but the model is prompted to generate errors as chain-of-thought (Wei et al., 2022) reasoning before each distractor. We note that Feng et al. (2024) use teacher-written feedback instead of errors as CoT, but we use errors for a fairer comparison to our method. We make other small updates to the prompts to include question tags and to generate either 3 or 10 distractors per question. All our prompts are shown in Supplementary Material H. In Supplementary Material B, we show results from other baselines in Feng et al. (2024) on our data. We also perform a preliminary, small-scale comparison with the recent OpenAI o1 model (OpenAI, 2024b), which reportedly performs better than GPT-4o on mathematical reasoning tasks.

**Fine-Tuning Baselines.** We introduce three strong fine-tuning baselines for distractor generation. The first is **DisSearch-D**, where we fine-tune the base LLM to directly generate a distractor from the question stem, i.e., training  $p(d|s)$  on all question-distractor pairs. The second is **DisSearch-ED CoT**, where we fine-tune the base LLM to generate the error first, as chain-of-thought (Wei et al., 2022) reasoning, followed by the distractor, i.e., training  $p(e, d|s)$  on all question-distractor pairs. At inference time, we also use beam search to match the overgenerate-and-rank setup used in DiVERT. The third is **DisSearch-ED CoT Pipeline**, where we use the fine-tuned  $p(d|s, e)$  and  $p(e|s)$  models without variational training and test with the same two-step pipeline as DiVERT.

<sup>2</sup><https://eedi.com/us>

Model	K=3			K=10		
	Exact@3	Partial@3	Prop@3	Exact@10	Partial@10	Prop@10
Proprietary Base LLM, GPT-4o						
GPT-4o Zero-shot CoT (Feng et al., 2024)	6.22 $\pm$ 2.18	69.14 $\pm$ 3.97	35.27 $\pm$ 1.74	19.47 $\pm$ 3.30	78.66 $\pm$ 3.71	50.00 $\pm$ 1.96
GPT-4o kNN (Feng et al., 2024)	<b>21.28</b> $\pm$ 3.43	<b>78.42</b> $\pm$ 5.81	<b>49.63</b> $\pm$ 3.88	33.47 $\pm$ 3.48	85.14 $\pm$ 4.77	60.19 $\pm$ 3.87
Open-source Base LLM, MetaMath-Mistral 7B						
DisSearch-D	13.74 $\pm$ 2.86	74.13 $\pm$ 4.52	41.76 $\pm$ 0.50	34.12 $\pm$ 2.33	86.17 $\pm$ 4.91	61.41 $\pm$ 2.68
DisSearch-ED CoT	<u>14.11</u> $\pm$ 1.22	73.18 $\pm$ 3.81	42.14 $\pm$ 1.52	<u>36.21</u> $\pm$ 1.21	<u>86.77</u> $\pm$ 3.80	<u>62.83</u> $\pm$ 2.52
DisSearch-ED CoT Pipeline	13.53 $\pm$ 1.74	73.63 $\pm$ 4.05	41.26 $\pm$ 1.87	32.97 $\pm$ 3.48	86.23 $\pm$ 3.88	60.42 $\pm$ 2.80
DiVERT (ours)	13.37 $\pm$ 1.70	<u>76.33</u> $\pm$ 4.33	<u>42.87</u> $\pm$ 2.45	<b>37.00</b> $\pm$ 3.29	<b>87.26</b> $\pm$ 4.29	<b>63.24</b> $\pm$ 3.37

Table 1: Cross-validation performance on distractor generation for all approaches across all metrics. DiVERT, using an open-source base LLM with 7B parameters, outperforms all baselines and performs on par with or better than the much larger and proprietary GPT-4o. Best performance is in **bold** and second best is underlined.

## 4.4 Experimental Setup

For the prompting baselines, we use the same implementation and hyperparameters as the public code repository in Feng et al. (2024) for a fair comparison. We use the latest base LLM from the GPT-4 family, GPT-4o (as of June 13, 2024), instead of GPT-3.5, to further strengthen their performance. For DiVERT, we use MetaMath-Mistral 7B (Yu et al., 2023) as our base LLM since it is one of the best-performing LLMs in the 7B family on mathematical reasoning; we found that it outperforms other open-source LLMs with similar size on our tasks. See Supplementary Material F for detailed parameter settings. At test time, we overgenerate  $N_e = 10$  errors via LLM<sup>e</sup> using diverse beam search (Vijayakumar et al., 2018) and  $N_d = 10$  distractors through LLM<sup>d</sup> using standard beam search. Among the set of  $N_e \times N_d = 100$  error-distractor pairs we select the top- $K \in \{3, 10\}$ . For DisSearch-ED CoT, we use standard beam search with 100 beams for a fair comparison with DiVERT. DisSearch-D performs better with 10 beams so we report its performance with 10 beams. For the primary results in Table 1, we perform a 5-fold cross-validation, rotating the test set over the dataset, and report the average and standard deviation of metrics across methods. For all other experiments, we evaluate on a single fold to reduce training time.

## 5 Results, Analysis, and Discussion

In this section, we quantitatively evaluate the quality of generated distractors, qualitatively evaluate both errors and distractors, perform error analyses on failed cases, and conduct an ablation study.

## 5.1 Quantitative Evaluation

**DiVERT performs comparably or better than GPT-4o and outperforms baselines.** Table 1 shows downstream distractor generation performance for all approaches on all evaluation metrics. Our variational approach, DiVERT, using an open-source base LLM with 7B parameters, performs on par with GPT-4o, especially on the robust Proportional@10 metric, where it outperforms GPT-4o by a wide margin. GPT-4o kNN performs best on  $K = 3$  evaluation metrics, which is not surprising since prior work (Feng et al., 2024) found that the kNN approach exploits in-context examples in the training set that have the same underlying structure as the target MCQ, different only in named entities and numerical values. Therefore, all it needs to do is to follow patterns and generate three distractors correspondingly without understanding errors. However, it does not perform as well on  $K = 10$  metrics since it often fails to go beyond the top three errors in MCQs that have numerous plausible errors. In contrast, DiVERT performs well by training on error labels to acquire an understanding of plausible mathematical errors. Under the same base LLM, DiVERT performs better than baselines on almost all metrics, which highlights the importance of its variational approach, since sampling from the LLM<sup>g</sup> model during training encourages exploration of the error space and improves model robustness.

**Errors as “chain-of-thought” improve performance.** Comparing among baselines, we see that training on human-authored error explanations improves distractor generation performance. This result can easily be explained by error labels serving as valuable chain-of-thought (Wei et al., 2022) supervision during training.

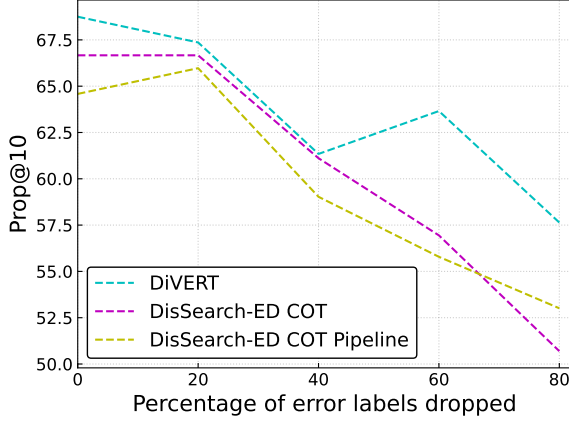


Figure 2: Distractor generation performance with increasing percentages of error labels dropped (unused in training). DiVERT outperforms baselines, especially when only a small number of error labels are used.

Model	Exact	Partial	Prop
DiVERT	<b>42.36</b>	<b>91.67</b>	<b>68.75</b>
– variational training	35.41	90.27	64.81
– train $q_\phi$ (freeze $q_\phi$ )	37.50	91.66	66.20
– $q_\phi$ regularization	40.97	89.58	66.20
– temperature annealing	41.66	89.58	67.36

Table 2: Ablation study of DiVERT on distractor generation performance on  $K = 10$  metrics on a single fold.

**DiVERT works even with a small number of error labels.** Since soliciting error labels behind question-distractor pairs from math educators is time-consuming, we investigate DiVERT’s reliance on the amount of error label data. Figure 2 shows DiVERT’s distractor generation performance using different portions of the available error labels. We see that on the most stable metric, Prop@10, DiVERT performs better than baselines, especially when the majority of error labels are dropped: the performance gap is widest when less than half of the training error labels are used for training. This result further highlights the importance of the self-exploration training setup in DiVERT’s variational approach at going beyond human-authored labels and linking different error types across questions.

**Ablation study.** Table 2 shows the results of the ablation study on the more robust @10 metrics on a single fold. We see that the variational training process is crucial and removing it results in a large drop in performance (see Supplementary Material C). We also see that not training  $q_\phi$  and using its fine-tuned initialized version  $q_{\phi_{init}}$  also drops performance, which highlights the importance of

Q: What is the lowest common multiple of 12 and 15?	
Error	Distractor
Ground-Truth	
Believes finding the product of two numbers gives their lowest common multiple.	180
Believes the largest number in a set of numbers is always their lowest common multiple.	15
Confuses factors and multiples.	3
DiVERT	
Believes they can find the lowest common multiple by adding the numbers together.	27
Confuses factors and multiples.	3
When asked for the lowest common multiple, thinks they can just give any multiple of one of the numbers.	15
GPT-4o kNN	
Confuses the product of the two numbers with the LCM.	180
Incorrectly identifies a common multiple that is not the lowest.	120
Incorrectly adds the two numbers.	27
GPT-4o Zero-shot CoT	
Confusing the concept of the lowest common multiple with the greatest common divisor.	3
Incorrectly adding the two numbers instead of finding the lowest common multiple.	27
Confusing the concept of multiples with that of prime factors.	5
DisSearch-ED CoT	
Confuses factors and multiples.	3
Does not understand the term multiple.	15
Identifies a common multiple but not the lowest common multiple.	75

Table 3: Examples of errors and corresponding distractors generated by different approaches for a test MCQ.

sampling from  $q_\phi$  and exploring the error space of math MCQs. Regularizing  $q_\phi$  and using temperature annealing in the soft token approximation are also effective at making training more robust.

## 5.2 Qualitative Evaluation

Table 3 shows generated errors and corresponding distractors for an MCQ stem in the test set. DiVERT and GPT-4o kNN each generate a diverse set of plausible errors and corresponding distractors. However, GPT-4o Zero-shot CoT prioritizes error diversity and generates errors unlikely to be made by real students like “Confusing the concept of multiples with that of prime factors”. The best fine-tuning baseline, DisSearch-ED CoT, generates generic error descriptions like “Does not understand the term multiple”, which is not specific to the question’s context.

**Failure Pattern Analysis.** We now investigate failure patterns in the generated errors and corresponding distractors from DiVERT. Table 4 shows a representative example MCQ in the test set. We observe that a majority of generated errors from  $p(e|s)$  are mathematically valid, with high diversity, but some are less likely among real students such as “Subtracts instead of divides”, matching the observation made in prior work (Feng et al., 2024). By far, the most frequent failure pattern we observe is on the controllable distractor generation model  $p(d|s, e)$ , where the generated distractor is not faithful/consistent to the error. In this example, for the error “When dividing a fraction by an integer, divides the denominator by the integer”, the generated distractor is  $\frac{6}{1}$  rather than the consistent  $\frac{6}{3}$ . The reverse can also occasionally happen when a distractor is plausible but the error is not. This observation highlights that although effectively learning good error representations and capable of identifying what errors can be made in a math MCQ, the biggest limitation of DiVERT is in its inability to enforce consistency in the downstream distractor generation model  $p(d|s, e)$ . This limitation suggests a major direction for future work, possibly by exploring the use of an error-distractor consistency penalty in the training objective.

## 6 Human Evaluation of Errors

### 6.1 Evaluation Setup

While automated distractor evaluation is well-defined since there is ground-truth, it is more challenging to automatically evaluate errors. Reference-based evaluation may penalize errors that faithfully reflect the mathematical error behind a distractor but are semantically different than the ground truth, or conversely, reward errors that are invalid but semantically similar to the ground truth. To address this challenge, we conduct a human evaluation to measure the quality of generated and human-authored errors. We recruit two experienced math teachers who have extensive expertise in designing math MCQs as annotators. We randomly select 20 questions from the test set on a diverse set of topics, and for each question, show annotators the ground-truth, human-authored errors, and generated errors from both DiVERT and GPT-4o, for a total of 180 errors. For DiVERT, we select the top 3 distractors from the  $p(e|s)$  model using diverse beam search to promote error diversity. For GPT-4o, we select the errors generated during CoT

Question stem: Calculate: $\frac{6}{9} \div 3$	
Error	Distractor
Plausible error, plausible and consistent distractor.	
When dividing a fraction by an integer, divides both the numerator and denominator by the integer.	$\frac{2}{3}$
Plausible error, plausible but inconsistent distractor.	
When dividing a fraction by an integer, divides the denominator by the integer.	$\frac{6}{1}$
Implausible error, plausible but inconsistent distractor.	
Divided by the denominator instead of the numerator.	$\frac{1}{3}$
Implausible error, implausible and inconsistent distractor.	
Subtracts instead of divides.	$\frac{3}{6}$

Table 4: Qualitative error analyses of generated errors and corresponding distractors from DiVERT on an example MCQ stem from the test set.

	Human	DiVERT	GPT-4o
Rating	$3.23 \pm 1.28$	$3.07 \pm 1.39$	$2.56 \pm 1.25$

Table 5: Average error quality rated by math teachers. Human and DiVERT errors are similar in quality, and both are better than GPT-4o with statistical significance.

distractor generation. We randomize the order of errors shown to annotators and do not tell them how each error is generated. We instruct annotators to rate each error on a 5-point Likert scale, with 5 being the best, depending on whether an error is relevant to the question, mathematically sound, specific, conceptual, and likely to be made by a real student. See Supplementary Material I.1 for the exact instructions given to annotators.

### 6.2 Results

Table 5 shows the average and standard deviation of annotators’ ratings on errors that were human-authored and LLM-generated, by both DiVERT and GPT-4o. We find that the quality of DiVERT’s errors are close to human ones, with no statistically significant difference between them using a Two-Sample  $t$ -Test ( $p = 0.36$ ). This result suggests that DiVERT retains the human-level quality of errors it is trained on during fine-tuning. Moreover, we find that both human and DiVERT-generated errors are better than GPT-4o ones, with statistical significance ( $p < 0.01$ ). This result is promising since our base LLM, MetaMath-Mistral 7B, is orders of magnitude smaller than GPT-4o. Qualitatively, we



find that GPT-4o’s errors are often not what real students are likely to make, and even occasionally confuse the correct solution approach with an error. This result shows that even state-of-the-art LLMs are not able to anticipate student errors, and that training on human-authored labels is likely necessary for LLMs to accurately diagnose student errors. Finally, we note that overall, the error labels vary a lot in terms of quality and score lower than expected. The Pearson correlation coefficient between our annotators’ ratings is also only 0.33, indicating low-to-moderate agreement. This result is due to many errors deemed to be unlikely to be made by students by annotators, rather than being mathematically incorrect. It is likely that even for humans, anticipating errors made by students is a challenging task, which suggests that future work should focus on understanding the nature of errors made by real students and their causes.

## 7 Related Work

For automated distractor generation in language learning and reading comprehension, prior works have explored ranking candidate distractors based on semantic similarity to the key and using knowledge graphs (Susanti et al., 2018; Stasaski and Hearst, 2017; Alsubait et al., 2014). More recent works use an end-to-end pipeline for distractor generation, which lead to longer and higher-quality distractors (Lee et al., 2024; Qiu et al., 2020; Shuai et al., 2023; Xie et al., 2021; Gao et al., 2019), also leveraging open-source LLMs such as BERT and T5 (Kalpakchi and Boye, 2021; Chiang et al., 2022; Rodriguez-Torrealba et al., 2022; Qu et al., 2024; Wang et al., 2023a); these approaches are similar to the baselines we use in this work. Other works prompt state-of-the-art proprietary LLMs such as ChatGPT and GPT-4 to generate distractors, with carefully crafted prompts, for a wider range of subjects including math and computer science (Tran et al., 2023; Bitew et al., 2023; Feng et al., 2024). Evaluating the quality of distractors, however, remains challenging: (Moore et al., 2024) proposes a series of features that can be used to evaluate the quality of distractors, although those features are mostly about surface semantics and do not apply to subjects like math that require deeper reasoning. Another line of recent work in (Feng et al., 2024; Scarlatos et al., 2024a) proposes to measure the quality of distractors through how likely they are going to be selected by real students, which

requires training another model to predict student option selection behavior. However, most existing MCQ datasets do not come with such student response information.

At a high level, an alternative to our approach of representing errors as text is to learn *latent* error representations (Li et al., 2020; Tu et al., 2022), i.e., characterizing errors as a latent stochastic error vector, possibly in the text embedding space. However, in our experiments, we found this approach to be uninterpretable and ineffective, significantly reducing distractor generation performance, possibly because errors behind distractors in math MCQs correspond to complex reasoning processes and are harder to capture than different user writing styles. Our approach in improving the diversity of generated errors also bears some resemblance to the works in (Wen et al., 2022; Wang et al., 2023b), although the nature of our task is more complex than dialogue generation and question answering.

## 8 Conclusions and Future Work

In this paper, we proposed DiVERT, a novel variational approach to jointly learn an interpretable, textual representation of errors in math MCQs and how to generate distractors that correspond to these errors. On a real-world math MCQ dataset, we showed that DiVERT results in better downstream distractor generation performance over state-of-the-art approaches. We also conducted a human evaluation with math educators and found that DiVERT leads to errors as interpretable as those provided by humans, and outperforms GPT-4o, despite using a much smaller base LLM.

There are many avenues for future work. First, we plan to apply the learned error representations to another downstream task, feedback generation, and explore whether our approach leads to a more accurate student profile/model, which will in turn result in higher quality feedback in math tutoring chatbots. Second, we plan to test the across-topic generalizability of our approach and investigate whether the error representations can generalize to previously unseen math topics. Third, we hope to develop new, LLM-based metrics to evaluate the mathematical validity of an error and the equivalence between two error explanations, to alleviate the need for human evaluation of error quality.

## Acknowledgements

The authors would like to thank the annotators and Jaewook Lee, Hunter McNichols, Hasnain Heickal, and Nancy Otero for helpful discussions. We also thank the anonymous reviewers for their helpful comments. This work is partially supported by Schmidt Futures under the learning engineering virtual institute (LEVI) initiative and the NSF under grants 2118706, 2237676, and 2341948.

## Limitations

We identify several technical and practical limitations of our work. First, the main limitation of DiVERT is its tendency to generate distractors that do not always correspond with the preceding generated error. However, we note that all baselines (including GPT-4) also exhibit this behavior, and we plan on addressing this limitation as a line of future work. Second, we note that DiVERT requires a set of human-authored errors for initialization through fine-tuning. This process of labeling distractors with textual errors can be time-consuming and difficult to scale, although we observe that DiVERT retains most of its ability when only a small subset of the data is labeled. Third, we only experiment with one dataset, since to the best of our knowledge there are no similar distractor datasets with labeled textual errors. Finally, we do not perform a human evaluation on the quality of the distractors themselves due to limited resources.

## Ethical Considerations

Our goal in this work is to develop a system that can automatically create distractors for assessments or practice problems, and do so in an explainable way so that the distractors can be easily verified by human educators. We hope that such systems will save educators time on content creation, allowing them to spend more resources on personal student interactions. However, there is a concern that such systems could replace human educator jobs, which is a shared concern across most domains with AI applications. We note that while there is little risk for bias in the creation of mathematical distractors, the use of textual errors introduces the possibility of generating biased text by inheriting tendencies from the base LLM. Another risk of automatically generating distractors is that lower-quality learning content compared to human-authored content could lead to negative student learning outcomes.

Because of these reasons, we recommend that generated errors and distractors be reviewed by experts before being deployed to real students.

## References

- Khan Academy. 2024. Khanmigo: Khan academy’s ai-powered teaching assistant. Online: <https://www.khanmigo.ai/>.
- 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 for multiple-choice questions: A survey of methods, datasets, and evaluation. *arXiv preprint arXiv:2402.01512*.
- Tahani Alsubait, Bijan Parsia, and Uli Sattler. 2014. Generating multiple choice questions from ontologies: Lessons learnt. In *OWLED*, pages 73–84. Cite-seer.
- Nischal Ashok Kumar, Nigel Fernandez, Zichao Wang, and Andrew Lan. 2023. [Improving reading comprehension question generation with data augmentation and overgenerate-and-rank](#). In *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*, pages 247–259, Toronto, Canada. 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. *arXiv preprint arXiv:2307.16338*.
- Shang-Hsuan Chiang, Ssu-Cheng Wang, and Yao-Chung Fan. 2022. Cdgp: Automatic cloze distractor generation based on pre-trained language model. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5835–5840.
- Neisarg Dave, Riley Bakes, Barton Pursel, and C Lee Giles. 2021. Math multiple choice question solving and distractor generation with attentional gru networks. *International Educational Data Mining Society*.
- Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2024. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36.
- Wanyong Feng, Jaewook Lee, Hunter McNichols, Alexander Scarlatos, Digory Smith, Simon Woodhead, Nancy Otero Ornelas, and Andrew Lan. 2024. Exploring automated distractor generation for math multiple-choice questions via large language models. In *Findings of the North American Chapter of the Association of Computational Linguistics (NAACL)*.

- Yifan Gao, Lidong Bing, Piji Li, Irwin King, and Michael R Lyu. 2019. Generating distractors for reading comprehension questions from real examinations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6423–6430.
- Aritra Ghosh, Beverly Woolf, Shlomo Zilberstein, and Andrew Lan. 2020. [Skill-based career path modeling and recommendation](#). In *2020 IEEE International Conference on Big Data (Big Data)*, pages 1156–1165.
- Neil T Heffernan and Cristina Lindquist Heffernan. 2014. The ASSISTments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, 24(4):470–497.
- 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*.
- Eric Jang, Shixiang Gu, and Ben Poole. 2017. [Categorical reparameterization with gumbel-softmax](#). In *International Conference on Learning Representations*.
- Jurenka et al. 2024. Towards responsible development of generative ai for education: An evaluation-driven approach. online: [https://storage.googleapis.com/deepmind-media/LearnLM/LearnLM\\_paper.pdf](https://storage.googleapis.com/deepmind-media/LearnLM/LearnLM_paper.pdf).
- Dmytro Kalpakchi and Johan Boye. 2021. Bert-based distractor generation for swedish reading comprehension questions using a small-scale dataset. *arXiv preprint arXiv:2108.03973*.
- Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.
- Tom Kubiszyn and Gary Borich. 2016. *Educational testing and measurement*. John Wiley & Sons, New Jersey, USA.
- Jaewook Lee, Digory Smith, Simon Woodhead, and Andrew Lan. 2024. Math multiple choice question generation via human-large language model collaboration. *International Educational Data Mining Society*.
- Chunyuan Li, Xiang Gao, Yuan Li, Baolin Peng, Xiujuan Li, Yizhe Zhang, and Jianfeng Gao. 2020. Optimus: Organizing sentences via pre-trained modeling of a latent space. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4678–4699.
- Chin-Yew Lin. 2004. [ROUGE: A package for automatic evaluation of summaries](#). In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.
- Xin Liu, Muhammad Khalifa, and Lu Wang. 2023. Bolt: Fast energy-based controlled text generation with tunable biases. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 186–200.
- Ilya Loshchilov and Frank Hutter. 2019. [Decoupled weight decay regularization](#). In *International Conference on Learning Representations*.
- Hunter McNichols, Jaewook Lee, Stephen Fancsali, Steve Ritter, and Andrew Lan. 2024. Can large language models replicate its feedback on open-ended math questions? *International Educational Data Mining Society*.
- Steven Moore, Eamon Costello, Huy A Nguyen, and John Stamper. 2024. An automatic question usability evaluation toolkit. *arXiv preprint arXiv:2405.20529*.
- Anthony J. Nitko. 1996. *Educational assessment of students*. Prentice-Hall, Iowa, USA.
- OpenAI. 2024a. [Hello gpt-4o](#).
- OpenAI. 2024b. [Introducing openai o1](#).
- Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.
- Vishakh Padmakumar and He He. 2024. [Does writing with language models reduce content diversity?](#) In *The Twelfth International Conference on Learning Representations*.
- Vijay Prakash, Kartikay Agrawal, and Syaamantak Das. 2023. Q-genius: A gpt based modified mcq generator for identifying learner deficiency. In *International Conference on Artificial Intelligence in Education*, pages 632–638. Springer.
- Zhaopeng Qiu, Xian Wu, and Wei Fan. 2020. Automatic distractor generation for multiple choice questions in standard tests. *arXiv preprint arXiv:2011.13100*.
- Fanyi Qu, Hao Sun, and Yunfang Wu. 2024. Unsupervised distractor generation via large language model distilling and counterfactual contrastive decoding. *arXiv preprint arXiv:2406.01306*.
- Rajkumar Ramamurthy, Prithviraj Ammanabrolu, Kianté Brantley, Jack Hessel, Rafet Sifa, Christian Bauckhage, Hannaneh Hajishirzi, and Yejin Choi. 2022. Is reinforcement learning (not) for natural language processing: Benchmarks, baselines, and building blocks for natural language policy optimization. *arXiv preprint arXiv:2210.01241*.



- Steven Ritter, John R Anderson, Kenneth R Koedinger, and Albert Corbett. 2007. Cognitive tutor: Applied research in mathematics education. *Psychonomic bulletin & review*, 14(2):249–255.
- Ricardo Rodriguez-Torrealba, Eva Garcia-Lopez, and Antonio Garcia-Cabot. 2022. End-to-end generation of multiple-choice questions using text-to-text transfer transformer models. *Expert Systems with Applications*, 208:118258.
- 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)*.
- Alexander Scarlatos, Digory Smith, Simon Woodhead, and Andrew Lan. 2024b. Improving the validity of automatically generated feedback via reinforcement learning. In *International Conference on Artificial Intelligence in Education*, pages 280–294. Springer.
- Chantal Shaib, Joe Barrow, Jiuding Sun, Alexa F Siu, Byron C Wallace, and Ani Nenkova. 2024. Standardizing the measurement of text diversity: A tool and a comparative analysis of scores. *arXiv preprint arXiv:2403.00553*.
- Pengju Shuai, Li Li, Sishun Liu, and Jun Shen. 2023. Qdg: A unified model for automatic question-distractor pairs generation. *Applied Intelligence*, 53(7):8275–8285.
- Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnnet: Masked and permuted pre-training for language understanding. *Advances in neural information processing systems*, 33:16857–16867.
- Shashank Sonkar and Richard G Baraniuk. 2023. Deduction under perturbed evidence: Probing student simulation (knowledge tracing) capabilities of large language models. In *Proceedings of the AIED Workshop on Empowering Education with LLMs – the Next-Gen Interface and Content Generation*.
- 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*, pages 303–312.
- Yuni Susanti, Takenobu Tokunaga, Hitoshi Nishikawa, and Hiroyuki Obari. 2018. Automatic distractor generation for multiple-choice english vocabulary questions. *Research and practice in technology enhanced learning*, 13:1–16.
- Ana Paula Tomás and José Paulo Leal. 2013. Automatic generation and delivery of multiple-choice math quizzes. In *Principles and Practice of Constraint Programming: 19th International Conference, CP 2013, Uppsala, Sweden, September 16-20, 2013. Proceedings 19*, pages 848–863. Springer.
- Andrew Tran, Kenneth Angelikas, Egi Rama, Chiku Okechukwu, David H Smith, and Stephen MacNeil. 2023. Generating multiple choice questions for computing courses using large language models. In *2023 IEEE Frontiers in Education Conference (FIE)*, pages 1–8. IEEE.
- Haoqin Tu, Zhongliang Yang, Jinshuai Yang, and Yongfeng Huang. 2022. Adavae: Exploring adaptive gpt-2s in variational auto-encoders for language modeling. *arXiv preprint arXiv:2205.05862*.
- Kurt VanLehn. 1982. Bugs are not enough: Empirical studies of bugs, impasses and repairs in procedural skills. *The Journal of Mathematical Behavior*.
- 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.
- Hui-Juan Wang, Kai-Yu Hsieh, Han-Cheng Yu, Jui-Ching Tsou, Yu An Shih, Chen-Hua Huang, and Yao-Chung Fan. 2023a. Distractor generation based on Text2Text language models with pseudo Kullback-Leibler divergence regulation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 12477–12491.
- Wenya Wang, Vivek Srikumar, Hannaneh Hajishirzi, and Noah A Smith. 2023b. Elaboration-generating commonsense question answering at scale. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1619–1635.
- Zichao Wang, Angus Lamb, Evgeny Saveliev, Pashmina Cameron, Jordan Zaykov, Jose Miguel Hernandez-Lobato, Richard E Turner, Richard G Baraniuk, Craig Barton, Simon Peyton Jones, et al. 2021. Results and insights from diagnostic questions: The neurips 2020 education challenge. In *NeurIPS 2020 Competition and Demonstration Track*, pages 191–205. PMLR.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.
- Yuqiao Wen, Yongchang Hao, Yanshuai Cao, and Lili Mou. 2022. An equal-size hard em algorithm for diverse dialogue generation. In *The Eleventh International Conference on Learning Representations*.
- 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, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. [Transformers: State-of-the-art natural language processing](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System*



*Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Jiayuan Xie, Ningxin Peng, Yi Cai, Tao Wang, and Qingbao Huang. 2021. Diverse distractor generation for constructing high-quality multiple choice questions. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:280–291.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2023. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

## A Monte Carlo Approximation of ELBO Training Objective

Since exact inference of the posterior distribution is computationally intractable,  $q_\phi(e|s, d)$  is an approximation of the true posterior. We perform approximate inference by maximizing the evidence lower bound (ELBO) on the observed data log-likelihood, given in Equation 2. The ELBO in Equation 2 can easily be approximated by Monte Carlo simulation, with samples drawn from  $q_\phi(e|s, d)$ , as shown below:

$$\begin{aligned} & \mathbb{E}_{q_\phi(e|s, d)}[\log p_{\theta_d}(d|s, e)] \\ & - \beta D_{\text{KL}}(q_\phi(e|s, d) \parallel p_{\theta_e}(e|s)) \\ & = \mathbb{E}_{q_\phi(e|s, d)}[\log p_{\theta_d}(d|s, e)] \\ & - \beta \mathbb{E}_{q_\phi(e|s, d)}[\log q_\phi(e|s, d) - \log p_{\theta_e}(e|s)] \\ & \approx \sum_{\hat{e} \sim q_\phi(e|s, d)} \log p_{\theta_d}(d|s, \hat{e}) - \beta \log q_\phi(\hat{e}|s, d) \\ & + \beta \log p_{\theta_e}(\hat{e}|s) \end{aligned}$$

## B Additional Baselines

In Table 6, we show results on distractor generation for additional baselines from (Feng et al., 2024), including kNN using feedback instead of errors and the rule-based method, where GPT-4o first selects errors from a pool before generating distractors. We also run the CoT baseline using the OpenAI o1 model (OpenAI, 2024b). To reduce costs, we collect these additional results on a single fold of the data, and show the other GPT-4o baselines and DiVERT performance on this fold for reference.

We first observe that CoT with o1 only slightly outperforms CoT with GPT-4o, improving Prop@10 by 1.62. This indicates that the long internal reasoning generated by o1 does not increase alignment with teacher-written distractors, though it may reduce arithmetic or logical errors resulting in the slight increase. We next observe that the Rule-Based method is slightly better than CoT, though still significantly worse than kNN, consistent with findings in (Feng et al., 2024). Finally, we observe there is little difference when using feedback instead of errors in kNN. This result validates our choice of using errors as textual reasoning before generating distractors, and also indicates that the exact form of the reasoning may not be significant in the context of generating aligned distractors.

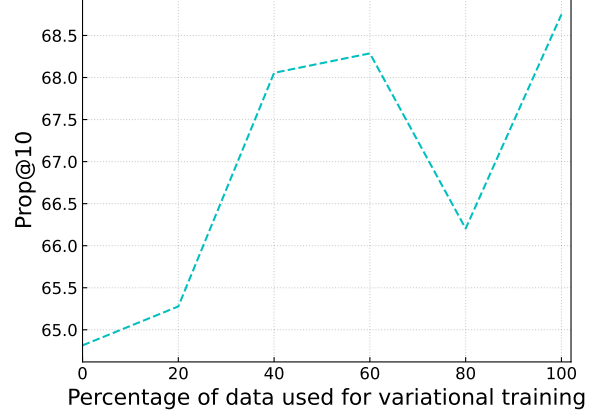


Figure 3: Distractor generation Prop@10 performance with an increasing percentage of data used for variational training. Sampling errors from  $q_\phi$  on all train question-distractor pairs performs best.

## C Variational Training Ablation: Q Model Helps DiVERT Learn a Robust Error Space

In real-world educational scenarios, there are multiple errors plausible among real students for the same MCQ stem. DiVERT, can learn a robust error space, by leveraging sampled errors from  $q_\phi(e|s, d)$  around the original human-authored ones during variational training. As shown in Figure 3, sampling errors from  $q_\phi$  on a higher number of question-distractor pairs in the train set, leads to a more robust learned error space, with better downstream distractor generation performance.

## D Real-world Math MCQ Dataset Details

We divide the real-world math MCQ dataset into train-val-test splits by *questions* to ensure no overlap in the MCQ stem across the splits, resulting in roughly a 72%-16%-12% split over question-distractor pairs. Table 7 shows detailed statistics of the dataset. Each of the 1434 MCQs has 3 distractors leading to  $1434 \cdot 3 = 4302$  question-distractor pairs. Among these pairs, 3601 pairs have human annotated error descriptions; other pairs are not labeled due to the non-mathematical nature of errors, e.g., careless slipping, reading the question incorrectly, etc. There are 1231 unique error descriptions. For the test set, we only keep questions containing human annotated error descriptions for all three associated distractors, leading to 144 questions with  $144 \cdot 3 = 432$  question-distractor pairs. We manually checked a random subset of the data and found no personally identifying information or

Model	K=3			K=10		
	Exact@3	Partial@3	Prop@3	Exact@10	Partial@10	Prop@10
Proprietary Base LLM, OpenAI o1						
OpenAI o1 Zero-shot CoT	4.17	72.22	36.11	25.69	77.78	52.08
Proprietary Base LLM, GPT-4o						
GPT-4o Zero-shot CoT (Feng et al., 2024)	2.78	68.75	33.33	20.83	81.25	50.46
GPT-4o Rule-Based (Feng et al., 2024)	4.17	67.36	35.42	26.39	77.78	53.70
GPT-4o kNN Feedback (Feng et al., 2024)	<b>25.69</b>	83.33	<b>55.09</b>	36.81	88.89	63.89
GPT-4o kNN Errors	23.61	<b>85.42</b>	52.78	36.81	88.89	63.66
Open-source Base LLM, MetaMath-Mistral 7B						
DiVERT (ours)	13.19	81.25	46.06	<b>42.36</b>	<b>91.67</b>	<b>68.75</b>

Table 6: Single fold performance on distractor generation for additional baselines and reference methods. Best performance is in **bold** and second best is underlined.

	Train				Validation				Test			
Math MCQ Dataset	2570 question-distractor pairs				599 question-distractor pairs				432 question-distractor pairs			
	$\mu$	$\sigma$	Min	Max	$\mu$	$\sigma$	Min	Max	$\mu$	$\sigma$	Min	Max
# tokens/Q stem	43.2	30.9	5	224	42.3	31.5	5	174	35.1	27.9	6	164
# tokens/solution	66.0	35.3	12	234	67.0	39.0	15	266	59.3	29.7	11	164
# tokens/key	9.3	11.5	1	189	9.5	14.6	1	175	7.5	4.7	1	44
# tokens/distractor	9.4	11.6	1	184	9.7	15.3	1	176	5.1	3.7	1	31
# tokens/error	14.3	6.0	4	42	14.0	6.1	5	36	13.8	5.5	5	39

Table 7: Statistics of the real-world math MCQ dataset which contains 1,434 MCQs across 41 unique subtopics.

offensive content.

## E Automated Error Evaluation

### E.1 Metrics

The open-ended and mathematical nature of errors makes automated text similarity metrics like ROUGE-L F1 (Lin, 2004) and BERTScore F1 (Zhang et al., 2020) unsuitable. Therefore, we conduct a **human evaluation** of generated errors, which we detail in Section 6. For completeness, we report error evaluation on automated metrics below.

We evaluate generated errors on two key aspects: 1) similarity with ground-truth, human-authored errors  $E$  with  $|E| = 3$ , and 2) diversity. We select the best 3 errors generated for each MCQ, i.e.,  $|\hat{E}| = 3$ . We compute both recall, which evaluates how well the generated errors recover actual human-authored errors, and precision, which evaluates how accurate the generated errors are with respect to the human-authored errors. Concretely, we measure recall by

$$\text{sim}_r^h(E, \hat{E}) = \sum_{e \in E} \max_{\hat{e} \in \hat{E}} (h(e, \hat{e})) / |E|$$

and precision by

$$\text{sim}_p^h(E, \hat{E}) = \sum_{\hat{e} \in \hat{E}} \max_{e \in E} (h(\hat{e}, e)) / |\hat{E}|,$$

where  $h$  denotes the choice of the textual similarity metric. We use traditional textual similarity metrics including **ROUGE-L F1** (Lin, 2004) and **cosine similarity** using the pre-trained SBERT encoder MPNet (Song et al., 2020), as well as recent metrics like **BERTScore F1** (Zhang et al., 2020).

For **diversity**, following (Padmakumar and He, 2024; Shaib et al., 2024), we report the complement of the homogenization score of a set of errors  $E$  as

$$\text{div}^h(E) = 1 - \sum_{e_1 \neq e_2 \in E} h(e_1, e_2) / |E \times E|,$$

where  $h$  denotes the choice of the textual similarity metric. We report diversity for both human-authored errors  $E$  and predicted errors  $\hat{E}$ , averaged across all test MCQs.

### E.2 Baselines and Results

We introduce a new fine-tuning baseline for error generation, **ErrorSearch-E**, where we fine-tune the base LLM to directly generate an error from the question stem, i.e., training  $p(e|s)$  on all error-question pairs. For a fair comparison, we generate errors from the  $p(e|s)$  model of DiVERT in a standalone fashion. We use diverse beam search decoding (Vijayakumar et al., 2018) to generate errors from both models. We also compare with errors generated from GPT-4o Zero-shot CoT, as well as

Model	ROUGE-L F1				BERTScore F1				Cosine Similarity			
	Precis.	Recall	F1	Div.	Precis.	Recall	F1	Div.	Precis.	Recall	F1	Div.
Proprietary Base LLM, GPT-4o												
GPT-4o Zero-shot CoT	0.251	0.272	0.261	0.718	0.632	0.635	0.633	0.293	0.602	0.607	0.605	0.376
Open-source Base LLM, MetaMath-Mistral 7B												
ErrorSearch-E	<u>0.498</u>	<b>0.597</b>	<u>0.543</u>	<u>0.781</u>	<u>0.741</u>	<b>0.786</b>	0.763	0.368	0.698	<b>0.759</b>	<b>0.727</b>	0.475
DisSearch-ED CoT	<b>0.595</b>	0.526	<b>0.558</b>	0.448	<b>0.786</b>	0.751	<b>0.768</b>	0.207	<b>0.735</b>	0.702	<u>0.718</u>	0.278
DiVERT $p(e s)$ (ours)	0.479	<u>0.576</u>	0.523	<b>0.786</b>	0.732	<u>0.775</u>	0.753	<b>0.372</b>	0.680	<u>0.746</u>	0.711	<b>0.487</b>

Table 8: Performance on automated error evaluation for all error-based approaches across all metrics. Best performance is in **bold** and second best is underlined.

the finetuning baseline DisSearch-ED CoT, both of which generate errors followed by distractors.

Table 8 shows error generation performance for all error-based approaches on all evaluation metrics. As a reference for the diversity of predicted errors shown, the diversity of ground-truth, human-written errors is 0.574, 0.300, and 0.349, for the choice of the similarity metric as ROUGE-L F1, cosine similarity, and BERTScore F1, respectively. The finetuning baseline ErrorSearch-E imitates the ground-truth human-written error distribution and performs best on recall performance across all textual similarity metrics. DisSearch-ED CoT performs best on precision performance across all textual similarity metrics. However, the same beam search decoding helping precision, leads to a drop in the performance of DisSearch-ED CoT on recall and diversity. GPT-4o Zero-shot CoT exhibits good error diversity, but as expected performs poorly on textual similarity metrics, with the zero-shot errors generated not matching the distribution of human-written errors. The  $p(e|s)$  model from our variational method, DiVERT, generates errors with the highest diversity. This diversity also leads to a slight drop in overall F1 performance across textual similarity metrics. This result is not surprising since by design, during the variational training of DiVERT, the  $p(e|s)$  model aligns with the entropy model  $q_\phi(e|s, d)$ , which is encouraged to generate error samples around human-written errors to learn a robust error space representation, leading to better downstream distractor generation performance.

We note that reference-based evaluation may penalize errors that faithfully reflect the mathematical error behind a distractor but are semantically different than the ground truth, or conversely, reward errors that are invalid but semantically similar to the ground truth. Therefore, we conduct a **human evaluation** of generated errors, which we detail in Section 6.

## F Experimental Setup

As detailed in Section 3.2, we finetune all three LLMs,  $LLM^e$ ,  $LLM^d$ , and  $LLM^q$ , using the collection of error label annotations behind question-distractor  $(s, d)$  pairs obtained from middle school math teachers, to initialize  $p_{\theta_e}(e|s)$ ,  $p_{\theta_d}(d|s, e)$ , and  $q_\phi(e|s, d)$ , respectively. We use the AdamW (Loshchilov and Hutter, 2019) optimizer with a batch size of 32, a learning rate of  $2e-5$ , and perform gradient clipping for training stability. We use the Parameter Efficient Fine-Tuning (PEFT) library from HuggingFace (Wolf et al., 2020) to load the base LLM, MetaMath-Mistral 7B, and train via low-rank adaptation (LoRA) (Hu et al., 2022) (LoRA  $\alpha = 256$ , LoRA  $r = 128$ , LoRA dropout = 0.05) using 8-bit quantization (Dettmers et al., 2024). We fine-tune for 5 epochs with early stopping on the validation set on a single NVIDIA A100 80GB GPU, with each epoch taking up to 35 minutes. We follow the same training setup for our finetuning baselines, DisSearch-D, DisSearch-ED, and DisSearch-ED CoT Pipeline.

After initialization, we perform DiVERT training for  $LLM^e$ ,  $LLM^d$ , and  $LLM^q$  using the same QLoRA setup as above. We use Monte Carlo simulation to approximate the ELBO in Equation 2 with 4 error samples drawn from  $q_\phi(e|s, d)$ . We use AdamW with a learning rate of  $5e-6$ , matching the learning rate in MetaMath finetuning (Yu et al., 2023), and perform gradient clipping for training stability. A single batch contains 16 question-distractor pairs, each having 4 Monte Carlo samples, for an effective batch size of 64. We set  $\beta$  in Equation 2 to 0.1, following prior work (Ghosh et al., 2020) to upweight the reconstruction loss. We set  $\alpha$  in Equation 4 to 0.95 to upweight the ELBO compared to the Q regularization loss. We train for 1 epoch on a single NVIDIA A100 80GB GPU, which takes up to 9 hours. Wherever possi-



ble, we use standard hyperparameters and do not do extensive parameter tuning like a grid search. Due to high computational and Open AI API cost, we report performance on one run of our DiVERT model and baselines. For metrics, for ROUGE we use the rouge-score library with Porter stemmer enabled, and for BERTScore we use the bert-score library with microsoft/deberta-xlarge-mnli as the underlying model. We additionally note that we used GitHub Copilot minimally in the writing of our code. All software we use in the development of this work is open source. We are consistent with the terms and intended use of all software and with the OpenAI API.

## G Example MCQs from Real-world Math MCQ Dataset

We show example MCQs from the dataset in Table 9.

## H Prompts

### H.1 Prompts for Base LLMs in DiVERT

We show all prompts used for the base LLMs in DiVERT, the error prior model  $p(e|s)$  parameterized by LLM<sup>e</sup> in Table 10, the controllable distractor generation model  $p(d|s, e)$  parameterized by LLM<sup>d</sup> in Table 11, and the error identifier model  $p(e|s, d)$  parameterized by LLM<sup>a</sup> in Table 12.

### H.2 Prompts for Prompting-based Baselines

We show all prompts used for prompting-based baselines, GPT-4o Zero-shot CoT in Table 13, and GPT-4o kNN in Table 14.

## I Human Evaluation Details

We received IRB approval for our human evaluation of error quality. Our evaluators were volunteers contacted through a research partner and were not compensated monetarily. They were made aware that their annotations would be used in scientific research in AI. We provide the instructions given to them for evaluating errors in Supplementary Material I.1.

### I.1 Human Evaluation Instructions

For each error, provide a rating between 1 and 5 (inclusive) in the “rating” column, where 1 is the worst and 5 is the best.

Please use the following criteria for evaluating errors:

- **Relevant:** The error should be applicable to the current question and the way it is solved.
- **Correct:** The error should be mathematically sound and concrete.
- **Specific:** The error should be specific to the question’s topic not be too generic.
- **Conceptual:** The error should be conceptual in nature, such that it could be applied to other similar questions.
- **Plausible:** The error should be likely to be made by some (or many) real students.
- **Overall Rating:** The rating you give should reflect the overall quality of the error across all the above criteria; you may deem that some criteria are more important than others depending on the context, so use your best judgment.

Question stem	James starts counting from $-2$ , adding one each time. What is the 5th number he says?
Topic	Adding and Subtracting Negative Numbers
Concept	Count forwards starting from a negative integer including through zero
Solution	2
Correct answer	Starting on $-2$ , we add one each time, moving up towards and then beyond 0 until we reach the 5th number, which is 2.
Distractor 1	6
Error 1	Counts on by 2, when asked to count forward in steps of 1
Distractor 2	3
Error 2	Counts on from the wrong number
Distractor 3	$-6$
Error 3	Counts on from the wrong number'
Question stem	$7^2 = ?$
Topic	Squares, Cubes, etc
Concept	Calculate the square of a number
Solution	$7^2 = 7 \times 7 = 49$
Correct answer	49
Distractor 1	14
Error 1	Mixes up squaring and multiplying by 2 or doubling
Distractor 2	72
Error 2	Reads a power as a normal digit
Distractor 3	77
Error 3	Mixes up squaring with repeating a digit
Question stem	What is the highest common factor of 8 and 28?
Topic	Factors and Highest Common Factor
Concept	Identify the Highest Common Factor of two numbers
Solution	8 has factors 1, 2, 4 and 8 and 28 has factors 1, 2, 4, 7, 14 and 28. The highest factor common to both is 4.
Correct answer	4
Distractor 1	28
Error 1	Believes the largest number in a set of numbers is always their highest common factor
Distractor 2	8
Error 2	Believes the smallest number in a set of numbers is always their highest common factor
Distractor 3	2
Error 3	Identifies a common factor but not the highest common factor

Table 9: Example MCQs from the real-world math MCQ dataset.

A teacher assigns the following math question to a class of middle school students.  
The question is: <question stem>  
The question topic is: <topic>  
The question concept is: <concept>  
The solution is: <worked out solution>  
The correct answer is: <answer>  
A possible error made by a student is:

Table 10: Prompt for error prior model  $p(e|s)$  parameterized by LLM<sup>e</sup> in DiVERT.

A teacher assigns the following math question to a class of middle school students.  
The question is: <question stem>  
The question topic is: <topic>  
The question concept is: <concept>  
The solution is: <worked out solution>  
The correct answer is: <answer>  
The error made by the student is: <error>  
The incorrect answer given by the student is:

Table 11: Prompt for controllable distractor generation model  $p(d|s, e)$  parameterized by LLM<sup>d</sup> in DiVERT.

---

A teacher assigns the following math question to a class of middle school students.  
The question is: <question stem>  
The question topic is: <topic>  
The question concept is: <concept>  
The solution is: <worked out solution>  
The correct answer is: <answer>  
The incorrect answer given by the student is: <distractor>  
The error made by the student is:

---

Table 12: Prompt for error identifier model  $q(e|s, d)$  parameterized by LLM<sup>q</sup> in DiVERT.

---

You are given the following math question along with the correct answer and explanation. Please use the following template to give <n> alternative incorrect answers to be used as multiple-choice options in a multiple-choice exam. Prior to the incorrect answer, provide the underlying error corresponding to that incorrect answer. These errors should be conceptual in nature and should not refer to numbers, variables, or names in the question.

[Template]  
Distractor1 Error:  
Distractor1:  
...  
Distractor<n> Error:  
Distractor<n>:

Question: <question>  
Topic: <topic>  
Concept: <concept>  
Explanation: <worked out solution>  
Answer: <answer>

---

Table 13: Prompt for GPT-4o Zero-Shot CoT.

---

You will be given a math question along with the correct answer and explanation. You will be also provided with several example questions that include incorrect distractor answers. Please generate <n> incorrect distractor answers for the current question to be used as multiple-choice options in a multiple-choice exam.

[Template]  
Distractor1 Error:  
Distractor1:  
...  
Distractor<n> Error:  
Distractor<n>:

<selected examples>

Question: <question>  
Topic: <topic>  
Concept: <concept>  
Explanation: <worked out solution>  
Answer: <answer>

---

Table 14: Prompt for GPT-4o kNN.