MATHWELL: Generating Educational Math Word Problems Using Teacher Annotations

Bryan R. Christ **Jonathan Kropko** **Thomas Hartvigsen**
University of Virginia
brc4cb@virginia.edu,jk8sd@virginia.edu,hartvigsen@virginia.edu

Abstract

Math word problems are critical K-8 educational tools, but writing them is time consuming and requires extensive expertise. To be educational, problems must be solvable, have accurate answers, and, most importantly, be educationally appropriate. We propose that language models have potential to support K-8 math education by automatically generating word problems. However, evaluating educational appropriateness is hard to quantify. We fill this gap by having teachers evaluate problems generated by LLMs, who find existing models and data often fail to be educationally appropriate. We then explore automatically generating *educational* word problems, ultimately using our expert annotations to finetune a 70B language model. Our model, MATHWELL, is the first K-8 word problem generator targeted at educational appropriateness. Further expert studies find MATHWELL generates problems far more solvable, accurate, and appropriate than public models. MATHWELL also matches GPT-4's problem quality while attaining more appropriate reading levels for K-8 students and avoiding generating harmful questions.[1]

1 Introduction

Math word problems (MWP) are natural language math questions paired with numerical answers and are critical tools for K-12 math education (Daroczy et al., 2015; Pearce et al., 2013; Schwartz, 2023; Verschaffel et al., 2020). Traditionally, teachers hand write MWPs customized to their students' interests, which has been shown to improve students' learning, test performance, and general interest in math (Bernacki and Walkington, 2018; Walkington, 2013; Walkington and Bernacki, 2019). However, teachers' time pressure is often so severe that they must use boilerplate question sets. We propose that large language models (LLMs) are poised to enhance math education by generating customized, diverse MWPs for students. Further, recent advances in math reasoning capabilities in LLMs may imply an approach towards educational impacts. However, it remains unknown whether math reasoning capabilities (Wei et al., 2023) translate to generating *educational* MWPs.

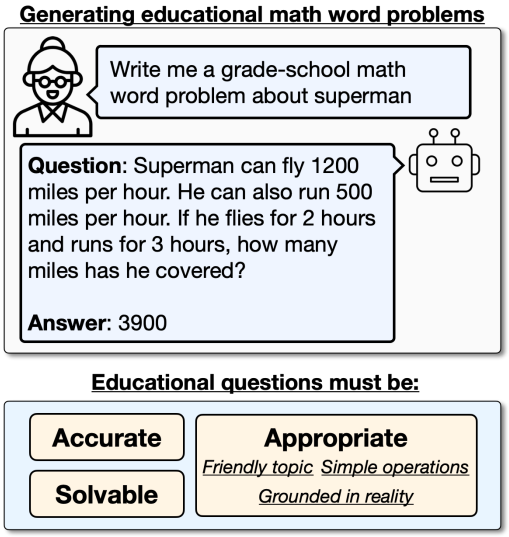

Figure 1: Generating educational math word problems with language models. To be educational, problems must simultaneously be solvable, accurate, and educationally appropriate.

We aim to explore and enhance LLMs' capacity to generate educational MWPs. There have been many works using traditional NLP methods to generate MWPs (Jiao et al., 2023; Koncel-Kedziorski et al., 2016; Niyarepola et al., 2022; Qin et al., 2023, 2024; Wang et al., 2021; Wu et al., 2022; Zhou and Huang, 2019; Zhou et al., 2023; Zong and Krishnamachari, 2023). However, they all rely on input *reference* MWPs or equations, ultimately largely rephrasing training data. To use them, teachers would need to manually curate problem sets, which is time consuming. And being traditional approaches, these methods are also not promptable, limiting personalization and MWP diversity.

[1] https://github.com/bryanchrist/MATHWELL

Findings of the Association for Computational Linguistics: EMNLP 2024, pages 11914–11938
November 12-16, 2024

Dataset	Example	Educational Appropriateness
GSM8K	Henry took 9 pills a day for 14 days. Of these 9 pills, 4 pills cost $1.50 each, and the other pills each cost $5.50 more. How much did he spend in total on the pills?	✗ This question mentions taking pills, which is not appropriate for young learners.
EGSM (Ours)	Barbie has 100 pink outfits. She has 20 more blue outfits than pink outfits. She has 50% more green outfits than blue outfits. How many outfits does Barbie have in total?	✓

Table 1: Example from both GSM8K and EGSM (ours). EGSM is the only teacher validated grade school math dataset, while existing datasets contain problems inappropriate for young learners.

Meanwhile, math reasoning in LLMs has been developing rapidly (Wei et al., 2023; Yao et al., 2023). However, to be useful in education demands strict, domain-specific criteria. Two recent works explore generating MWPs with LLMs (Niyarepola et al., 2022; Zong and Krishnamachari, 2023); however, they generate problems without answers.

To evaluate the educational quality of LLMs for math, we recruit real math teachers to assess MWPs generated by SOTA LLMs. As part of a large human evaluation study, these domain experts spend almost 100 hours evaluating three criteria that make MWPs educational: *Solvability*, *Accuracy*, and, most importantly, *Educational Appropriateness*. The nuance of appropriateness motivates the need for human evaluations, as it takes years of experience for teachers to develop this sense. Through experiments evaluating existing math reasoning datasets and MWPs generated from five existing LLMs, both public and private models, the teachers identify clear failings in educational appropriateness, especially from open models.This is somewhat unsurprising, as common math reasoning datasets, like GSM8K (Cobbe et al., 2021), are crowd-sourced and not intentionally educational. However, this may imply we should temper our expectations for direct use of LLMs for elementary math education.

Given a lack of appropriateness in existing datasets' MWPs, we generate the first teacher annotated, educationally appropriate MWP dataset and use it to finetune a new 70B LLM specifically for educational MWP generation. Experts find that our open model, MATHWELL, matches GPT-4 in educational MWP generation and is 40% better than the next best open model, generating MWPs that are simultaneously solvable, accurate, and appropriate 74% of the time. This performance demonstrates the value of domain expert involvement in developing LLMs for education. We also find that MATHWELL can be prompted to discuss topics customized to student interests (*e.g.,* Superman) and incorporate specific math operations, outputs MWPs with more appropriate reading levels, and produces fewer harmful errors than other models. We release our entire human evaluation, including over 5,000 MWPs with gold labels for key educational criteria, along with MATHWELL's training code and weights.

Our key contributions are as follows:

- We find existing math datasets are not sufficient to enhance the educational quality of LLM-generated MWPs.
- We collect over 5,000 annotations from real teachers, which we filter to create a new, large synthetic training dataset for educational MWP generation, Educational Grade School Math (EGSM).
- We use EGSM to finetune a performant LLM for educational MWP generation, which we release alongside our entire human study.

2 Related Work

Math QA Datasets There are many datasets for training and evaluating LLMs on grade school math reasoning. Several popular datasets include GSM8K (Cobbe et al., 2021), NumGLUE (Mishra et al., 2022), GSM-Hard (Gao et al., 2023), ASDIV (Miao et al., 2020) and SVAMP (Patel et al., 2021). Recent work has also developed new grade school math evaluation datasets or large training datasets with different solution rationales for existing datasets, particularly GSM8K, including both human written (Kim et al., 2023; Mishra et al., 2023) and synthetic (Mitra et al., 2024; Shi et al., 2023; Toshniwal et al., 2024; Yu et al., 2023; Yuan et al., 2023; Yue et al., 2023) data.

However, because existing datasets are designed primarily for training and evaluating grade school math reasoning, they are not aligned with training educational grade school MWP generators. Grade school MWP training data for educationally appropriate generators must contain high-quality grammar, be written at an appropriate mathematical difficulty and reading level for K-8 students, include questions similar to those students would encounter in the classroom, and be comprised of questions that are appropriate for an education setting. Further, to encourage effective solution generation, we use Program of Thought (PoT) solutions written as Python functions (see Appendix C.7 for details), as PoT outperforms Chain of Thought (CoT; Wei et al. 2023) for open-ended questions, which are the type of problems we seek to generate (Azerbayev et al., 2023; Gao et al., 2023; Yue et al., 2023).

We consider high-quality grammar and similarity to word problems students encounter in the classroom baseline criteria any potentially relevant training data must possess. Existing datasets that contain these baseline characteristics and are aligned with one or more of the other four criteria are GSM-Hard (Gao et al., 2023), GSM8K (Cobbe et al., 2021), MathInstruct GSM8K (Yue et al., 2023), ASDIV (Miao et al., 2020), and SVAMP (Patel et al., 2021), although none of these datasets contain all four criteria.

MWP Generation Other work explores automatically generating MWPs but requires reference problems or equations as model input and, therefore, constrains the diversity and range of possible outputs and largely rephrases training data. Closest to our work is Zong and Krishnamachari (2023), who assess GPT-3's ability to generate MWPs. Their method is reference-dependent, however, as they use a reference problem to guide generation. Further, they only generate MWPs, not solutions.

Most other works use LLMs or deep neural networks to generate MWPs based on pre-specified equations, provide additional MWPs based on reference problems, or re-write existing MWPS (Jiao et al., 2023; Koncel-Kedziorski et al., 2016; Norberg et al., 2023; Niyarepola et al., 2022; Qin et al., 2023, 2024; Wang et al., 2021; Wu et al., 2022; Zhou and Huang, 2019; Zhou et al., 2023), each of which restricts the range of possible outputs. These approaches require additional input from users, which is infeasible for teachers or students who wish to create customized MWPs, making them incomparable to our work. To address this issue, we generate MWP question/answer pairs simultaneously without a reference problem or equation, which we call reference-free generation. To the best of our knowledge, our study is the most comprehensive in exploring the educational appropriateness of generated MWPs alongside teachers.

3 Methods

Math word problems (MWPs) are natural language questions paired with numerical answers. We study generating these pairs with LLMs, prompting them to write new problems without augmenting hand-picked equations or reference problems. We evaluate these problems based on both the human and automatic evaluation criteria defined below.

Human Evaluation Criteria For model-generated MWPs to be educational, they must meet three criteria: 1) *Solvability*, where questions are possible to solve and have one correct answer. 2) *Accuracy*, where generated answers must be correct. 3) *Appropriateness*, where MWPs should be questions teachers would feel comfortable giving to K-8 students. Generally, appropriate MWPs should make sense, avoid grammatical errors or conflicting information, and be about topics and include mathematical operations appropriate for K-8 students in a school setting. Because this is hard to define, we emphasize real teacher evaluations of generated MWPs. Ideal MWPs are accurate, solvable, and appropriate, thereby meeting all criteria. We refer to such problems as MaC.

Automatic Evaluation Criteria Reading level automatically assesses if MWPs are written appropriately. Like Norberg et al. (2023), we use Flesch-Kincaid Grade Level (FKGL) to evaluate reading level. FKGL is a function of the total words, sentences, and syllables in a text, and the score represents a U.S. grade level (Aggarwal; Flesch, 1948; Kincaid et al., 1975). Thus, a FKGL score above 8 for a MWP would be considered inappropriate for an educational K-8 MWP generator. Lower reading levels for MWPs are preferred because high reading levels are known to harm student performance, especially for students who are already struggling (Walkington et al., 2018). Negative FKGL scores are possible and denote text that is easy to read due to having short words and sentences. We also calculate each MWP's average token length. Longer

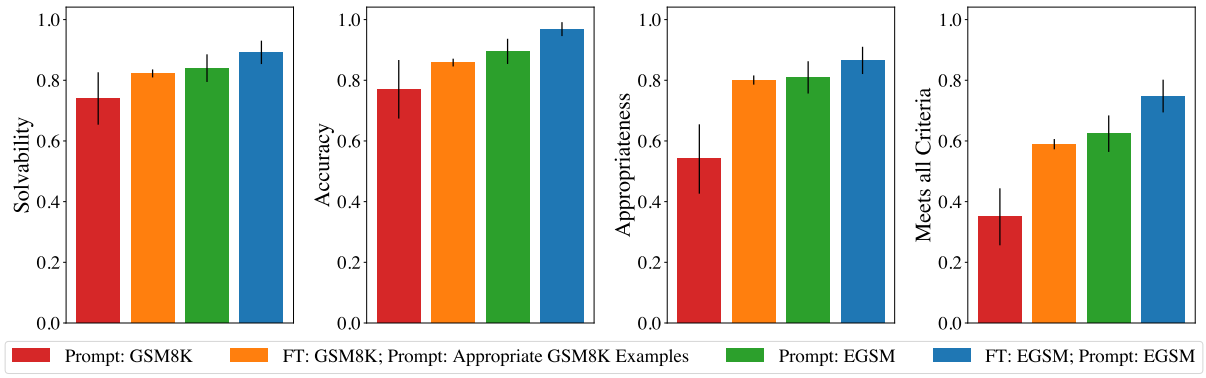

Figure 2: Llama-2 (70B) performance with 95% confidence intervals on our human evaluation metrics under different prompting/training scenarios. FT is supervised finetuning.

token length can be a proxy for MWP complexity, as longer questions tend to include more mathematical operations.

We follow prior works in using two automatic metrics to compare the quality of our synthetic MWPs to human-written MWPs: Perplexity (PPL) and BERTScore (Jiao et al., 2023; Zhou et al., 2023). Lower PPL implies better outputs and we calculate PPL using Llama-2 (70B). To show that PPL is not biased towards Llama-2 outputs, we report GPT-2 PPL in Table 12, finding the same trends in PPL discussed in the sections below. We use BERTScore (Zhang et al., 2020) to compute the semantic similarity of our synthetic MWPs and compare it to existing datasets. A lower BERTScore for synthetic MWPs relative to existing datasets would imply they are less similar to each other than human-written MWPs, while a higher score would imply they are more similar. In Section 4, we also use BERTScore to compare our synthetic MWPs to GSM8K to identify if they are similar to human-written MWPs. We calculate GSM8K's within-dataset BERTScore as a reference to compare each source against.

3.1 Evaluating Existing Datasets

We first aim to assess the degree to which existing datasets can be used to prompt models to generate educational K-8 MWPs. We focus our evaluation on GSM8K (Cobbe et al., 2021) since it is popular and high quality. For each generation, we randomly sample 8 MathInstruct GSM8K samples (Yue et al., 2023)[2] and use them to few-shot prompt Llama-2 (70B) with a standard prompt asking the model to generate a grade school MWP using PoT to compute numerical answers (prompting details in Appendix A). We randomly sample 100 generations with executable PoT solutions and acquire teacher annotations for solvability, accuracy, and appropriateness (we discuss further annotation details below). The size of this evaluation sample is consistent with the human evaluation samples used in other MWP generator studies (Jiao et al., 2023; Koncel-Kedziorski et al., 2016; Niyarepola et al., 2022; Qin et al., 2024; Wu et al., 2022; Zhou and Huang, 2019; Zhou et al., 2023; Zong and Krishnamachari, 2023). As shown in Figure 2 (red), we find that existing data is ill-suited for prompting models to generate educational word problems, with the worst performance being in appropriateness where barely over 50% of generations are appropriate, leading to only 35% of the generations being labeled as MaC. This finding suggests that educationally inappropriate samples such as the one shown in Table 1 are prevalent in GSM8K.

3.2 Expert Annotation

To address the educational inappropriateness of existing math datasets, we generate a high-quality, educationally-appropriate dataset for training MWP generators. Broadly, we generate synthetic data and evaluate it with teachers (see Figure 3 for our data generation process). To generate this data, we first finetune Llama-2 (70B) (Touvron et al., 2023) using the public MathInstruct GSM8K dataset (Yue et al., 2023) and QLoRA (Dettmers et al., 2023) (details in Appendix D). We next identify educationally appropriate GSM8K examples in consultation with teachers and use them to prompt our finetuned model to generate new grade school MWPs. We then acquire teacher annotations for solvability, accuracy, and appropriateness. All annotators were seasoned educators, with an average

[2]This dataset adds PoT solutions to GSM8K questions.

Dataset	N	PoT	Python Function	Appropriate Difficulty	Teacher Annotated	Length	Reading Level ↓	BF1
GSM-Hard	1,319	✓	✓	✗	✗	72.9 (25.6)	4.21 (2.43)	84.0
GSM8K	8,792	✗	✗	✓	✗	67.0 (24.4)	4.24 (2.47)	84.5
MathInstruct GSM8K	6,403	✓	✗	✓	✗	66.2 (23.9)	4.25 (2.48)	84.6
ASDIV	2,305	✗	✗	✓	✗	45.1 (15.8)	3.56 (2.40)	85.5
SVAMP	1,000	✗	✗	✓	✗	47.3 (11.7)	3.39 (2.07)	86.1
EGSM (Ours)	2,093	✓	✓	✓	✓	57.2 (15.7)	**2.50 (1.76)**	85.2

Table 2: Characteristics of datasets with more than 1,000 examples that can be used to train reference-free grade school MWP generators. N is the deduplicated number of questions, Length is average question length (in tokens), readability is measured by Flesch-Kincaid grade level, and BF1 is BERTScore F1. Standard deviations, where applicable, are in parentheses.

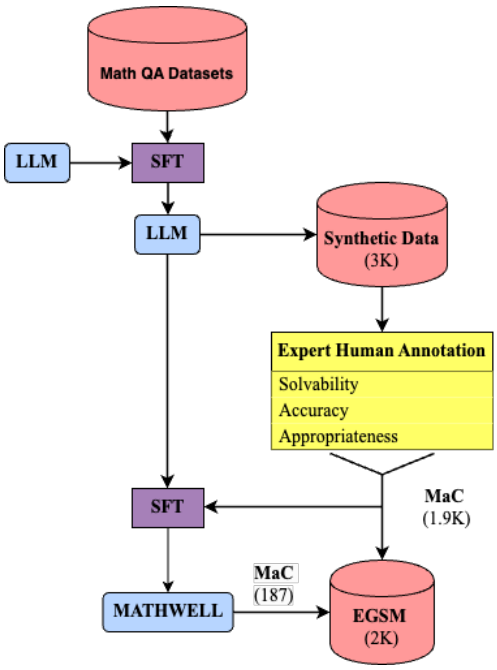

Figure 3: MATHWELL training and EGSM generation process. SFT is supervised finetuning and MaC denotes outputs that meet all criteria.

of 5.75 years of experience. They spent an average of 42 seconds per question, totaling 55.1 hours of expert annotation. In total, teachers annotated 3,234 synthetic MWPs, with 998 being annotated by two people and 232 annotated by three people.

Annotators agreed on solvability 84.6 ± 2.0% of the time, accuracy 92.0 ± 1.5% of the time, appropriateness 74.6 ± 2.4% of the time, all three labels 66.3 ± 2.6% of the time, and MaC 76.1 ± 2.4% of the time. The agreement rates for accuracy and solvability are higher than reported in recent human evaluation studies that analyze human preferences in LLM outputs, and the agreement rates for appropriateness and MaC are on par with these studies (Bai et al., 2022; Ouyang et al., 2022; Stiennon et al., 2022; Ziegler et al., 2020). As a result, we feel confident in the quality of our labels.

For handling disagreement, if the question was reviewed by two annotators and they disagreed on one of the criteria, we labeled the example as not having the desired criteria. If the question was reviewed by three annotators and there was a disagreement on one of the criteria, we assigned the label with the majority vote. Further details about the annotation process and annotator directions are in Appendix B.

3.3 Further Finetuning on High-quality Outputs

Based on this annotation process, we identified 1,906 MWPs that are simultaneously solvable, accurate, and appropriate, or meet all criteria (MaC). As shown in Figure 2 (orange), the initial finetuning and prompting using educationally appropriate data improves model performance, especially for appropriateness, but still less than 60% of generations meet all criteria. Therefore, to further improve performance and validate the quality of our synthetic data, we conduct additional finetuning on these MaC outputs to create a model we call MATHWELL. We compile MATHWELL's MaC outputs into a new dataset we call Educational Grade School Math (EGSM).

In total, EGSM contains contains 2,093 MaC question/answer pairs verified by teachers after the first and second stage of finetuning. To support research in automatically scoring model-generated MWPs, we release all our annotated data. To the best of our knowledge, this is the only teacher annotated MWP dataset (see Appendix B.5 for annotated data details). As shown in Figure 2 (green and blue), simply few-shot prompting with EGSM outperforms initial finetuning on GSM8K and the

additional finetuning on MaC outputs leads to further improvements, showing the second stage of finetuning is critical for educational alignment (see Appendix C.1 for more details on this ablation).

EGSM Dataset Characteristics Table 2 shows the advantages of EGSM over other datasets that have grammatically-correct questions similar to those students encounter in the classroom and are PoT, have Python function solutions, are written at an appropriate difficulty for K-8 students, and/or are educationally appropriate. Critically, EGSM is the only dataset annotated by teachers, ensuring its questions are appropriate for students, which Figure 2 shows leads to concrete improvements in human evaluation criteria. Second, EGSM has the lowest average reading level (evaluated by FKGL), so the questions may be more appropriate for those who struggle to read. Third, EGSM is the only dataset with both Program of Thought (PoT) solutions written as Python functions and questions that are mathematically appropriate for K-8 students, which we find critical for training reference-free MWP generators (see Appendix C.7).

EGSM is similar to existing datasets on other common evaluation metrics. While EGSM has a shorter average token length than GSM8K (Cobbe et al., 2021), its length is longer than ASDIV (Miao et al., 2020) and SVAMP (Patel et al., 2021). This suggests that EGSM's MWP complexity, as reflected in average length, is similar to existing data. EGSM's BERTScore is close to those of existing datasets, suggesting the MWPs are similar. Thus, EGSM is similar to human-written data while being more aligned with educational use.

4 Evaluating MATHWELL

Next, we thoroughly evaluate MATHWELL's generated MWPs using both human and automatic evaluations. We compare 250 randomly-selected MATHWELL outputs to those of open and closed source models. For closed-source models, we evaluate GPT-3.5 Turbo and GPT-4 Turbo. GPT-3.5 represents a closed-source model with similar capability to Llama-2 (70B) (Touvron et al., 2023), while GPT-4 is a much more capable model (OpenAI et al., 2024). For open-source models, we evaluate LLEMMA (34B) (Azerbayev et al., 2023) and MAmmoTH (70B) (Yue et al., 2023), which have similar capabilities to Llama-2 (70B) but are math specific. Llama-2 serves as a baseline to ensure task-specific finetuning improves performance. We prompt each model using examples from EGSM and ask it to create a question/answer pair on a random topic K-8 students are interested in (see Appendix F for a full list of topics). Our 250 evaluation sample for each model is roughly 2.5-5 times larger than the human evaluation samples used in other MWP generator studies (Jiao et al., 2023; Koncel-Kedziorski et al., 2016; Niyarepola et al., 2022; Qin et al., 2024; Wu et al., 2022; Zhou and Huang, 2019; Zhou et al., 2023; Zong and Krishnamachari, 2023).

[3]We use a smaller sample size (382) to assess executable code for GPT-4 due to the cost of querying its API.

4.1 Human Evaluation

MATHWELL Matches SOTA in Human Evaluation Criteria Teachers scored the 250 MWPs from each model for solvability, accuracy, and appropriateness (see examples in Appendix E and additional annotation details in Appendix B). Annotators also assessed topic specificity, or if the MWP incorporates the specified topic. Table 3 shows MATHWELL performs best among open-source models in all metrics, with a 19.9% higher share of outputs that MaC, 19.9% higher share that have executable code, and 43.4% higher share that have executable code and MaC than the next best open-source model. MATHWELL also outperforms GPT-3.5 in human evaluation criteria and matches 94.9% of GPT-4's performance in MaC, outperforming it in accuracy and appropriateness.

Although MATHWELL performs best in topic specificity, the other models do well in this metric, implying that LLMs can effectively customize MWPs to student interests. Table 3 also highlights that the open-source models lag significantly behind GPT-4 in human evaluation criteria, but that QLoRA finetuning on EGSM is enough to match its performance. Lastly, Table 3 highlights that open-source models have a large gap in the share of question/answer pairs that contain executable code relative to GPT-3.5, the best performing model in this metric (see Appendix G for details on GPT-4's performance on this metric).

MATHWELL Generates High-quality, Complex Questions MWPs involving multiplication, division, fractions, and decimals are more complex than those involving addition and subtraction, and we aim to train a generator capable of creating MWPs from each of these operations K-8 students regularly encounter. In Table 4, we assess whether

	Model	Solv.	Acc.	App.	MaC	Top. Spec.	EC	EC/MaC
API	GPT-4 Turbo	94.8 (1.41)	95.8 (1.31)	84.4 (2.36)	78.8 (2.59)	99.2 (0.56)	66.8 (2.41)	52.6 (1.73)
	GPT-3.5 Turbo	88.0 (2.06)	89.5 (2.07)	75.5 (2.91)	62.8 (3.06)	98.8 (0.69)	97.5 (0.32)	61.3 (2.99)
Public	LLEMMA	48.8 (3.17)	63.9 (4.37)	41.8 (4.48)	15.2 (2.28)	94.8 (1.41)	24.3 (0.70)	3.70 (0.55)
	MAmmoTH	86.8 (2.15)	94.9 (1.49)	67.7 (3.18)	56.8 (3.14)	97.6 (0.97)	6.90 (0.36)	3.91 (0.22)
	Llama-2	84.0 (2.32)	89.5 (2.12)	81.0 (2.72)	62.4 (3.07)	99.2 (0.56)	55.4 (0.98)	34.6 (1.70)
	MATHWELL (Ours)	**89.2 (1.97)**	**96.9 (1.17)**	**86.5 (2.29)**	**74.8* (2.75)**	**99.6 (0.40)**	**66.4* (1.00)**	**49.6* (1.83)**

Table 3: Comparing LLMs for MWP generation. All metrics are averages over 250 generations per model for human annotated criteria and over 2,000 for assessing the share of questions with executable code (EC).[3] Solv., Acc., App., MaC, Top. Spec., and EC/MaC are solvability, accuracy, appropriateness, meets all criteria, topic specificity, and the estimated share of questions that MaC and have executable code, respectively. Bold indicates the best open-source performance in each metric and a * indicates the difference between the best open-source performance and second open-source best performance is statistically significant at the p<.01 level. Standard errors are in parentheses.

		Solvable Questions							MaC Questions						
	Model	Add.	Sub.	Mult.	Div.	Frac.	Dec.	No Ops	Add.	Sub.	Mult.	Div.	Frac.	Dec.	Total Ops
API	GPT-4 Turbo	53.2	44.7	61.6	25.3	0.80	4.64	1.27	54.3	45.2	59.9	25.4	1.02	3.05	1.89
	GPT-3.5 Turbo	35.3	28.5	44.3	36.2	3.17	23.1	1.81	36.3	29.9	39.5	35.7	3.18	21.0	1.66
Public	LLEMMA	34.4	27.0	33.6	20.5	6.56	15.6	15.6	36.8	39.5	31.6	15.8	2.63	13.2	1.39
	MAmmoTH	39.6	37.8	43.8	19.4	3.69	10.6	2.30	43.0	42.2	40.8	16.9	4.93	9.86	1.58
	Llama-2	57.6	58.6	22.9	14.3	8.10	11.4	4.76	59.6	60.3	24.4	12.8	5.77	8.97	1.72
	MATHWELL (Ours)	69.5	69.1	24.7	10.3	5.38	7.62	1.35	71.1	70.6	24.6	8.56	4.81	7.49	1.87

Table 4: Characteristics of model-generated questions. Add., Sub., Mult., Div., Frac., Dec., No Ops, Total Ops, and MaC are addition, subtraction, multiplication, division, fractions, decimals, no operations, total operations, and meets all criteria, respectively. All columns are percentages except total ops, or the average number of distinct operations per question.

	Model	Strange	Too Hard	Harmful	Syntax	No Ops
API	GPT-4 Turbo	55.3	13.2	13.2	10.5	7.89
	GPT-3.5 Turbo	60.7	17.9	12.5	1.79	7.14
Public	LLEMMA	41.1	24.7	6.85	1.37	26.0
	MAmmoTH	77.9	7.79	5.19	2.60	6.49
	Llama-2	65.1	4.65	4.65	2.33	23.3
	MATHWELL	77.4	0.0	0.0	12.9	9.68

Table 5: Classification of appropriateness errors for each model. See Figure 8 for directions presented to annotators and Appendix E for inappropriate samples.

each model can generate MaC questions when using more complicated operations. Questions may contain more than one operation. Solvable questions may require no math operation if they contain the answer in the question (see Appendix E.6.3 for an example), so we also report the share of questions containing no operations. To determine if questions with complex operations are accurate and appropriate, we compare the math operations in solvable questions to those in MaC questions. We also report the average number of distinct operations in MaC questions for each model, which is another way to assess question complexity.

Table 4 shows that GPT-4 and MATHWELL are the only models for which the share of MaC questions for each operation is within two percentage points of that for solvable questions, showing MATHWELL can generate MaC questions regardless operation complexity. GPT-4 and MATHWELL are also the least likely to generate problems that require no operations and have the highest average total operations, which also suggests MATHWELL generates high-quality, complex problems. These findings suggest MATHWELL can effectively generate MWPs that cover the full range of problem types K-8 students encounter.

We do not prompt models for specific operations. Under these conditions, MATHWELL generates more problems containing addition and subtraction relative to the other models. In turn, there is a concern that MATHWELL's performance in Table 3 could be due to it generating simple questions for this experiment, which may be more likely to MaC. To address this concern, we conduct two additional analyses reported in Appendix C: 1) logistic regressions showing MATHWELL's higher MaC

relative to the other models except GPT-4 remains statistically significant when controlling for math operations and 2) a summary of accuracy by operation showing GPT-4 and MATHWELL are the only models for which accuracy does not substantially differ by operation and remains above 90% for each operation. These results buttress our finding that MATHWELL can generate MaC MWPs regardless of operation and pinpoint the operations for which each comparison model most commonly fail.

MATHWELL Makes Less Severe Appropriateness Errors As shown in the annotator directions in Figure 8, there are four primary reasons why a question would be flagged as inappropriate: being strange or unrealistic, being too difficult for a K-8 student, containing inappropriate content for a classroom setting, or having grammatical errors or typos. A final reason why a question would be labeled as inappropriate is that it does not require any mathematical operations to solve and, therefore, is a reading comprehension question rather than a MWP. See Appendix E for examples of each. Of these errors, questions that contain inappropriate content or are too difficult are the most harmful for young learners. Questions that are strange/unrealistic, have typos, or assess reading comprehension rather than math reasoning are not good, but they do not present a harm to student learning. As shown in Table 5, MATHWELL is the only model that does not make these more harmful errors, showing that it is better suited for generating MWPs that are educationally appropriate for young learners.

4.2 Automatic Evaluation

MATHWELL Outputs Have More Appropriate Readability Figure 4 compares the FKGL distribution of MATHWELL outputs to those of the other models considered. As shown in the figure, MATHWELL has the lowest FKGL and is the only model that does not generate questions beyond an 8th grade reading level, providing evidence it creates more readable questions for K-8 students. In Figure 9, we see a similar trend in readability for EGSM: Fewer of EGSM's MWPs than other existing datasets have a FKGL score greater than 8.

MATHWELL Outputs Are Human Quality Table 6 shows EGSM has lower PPL than GSM8K and Table 7 shows MATHWELL's 250 evaluation sample for the experiments in Table 3 has lower PPL than that of the other models. As shown in

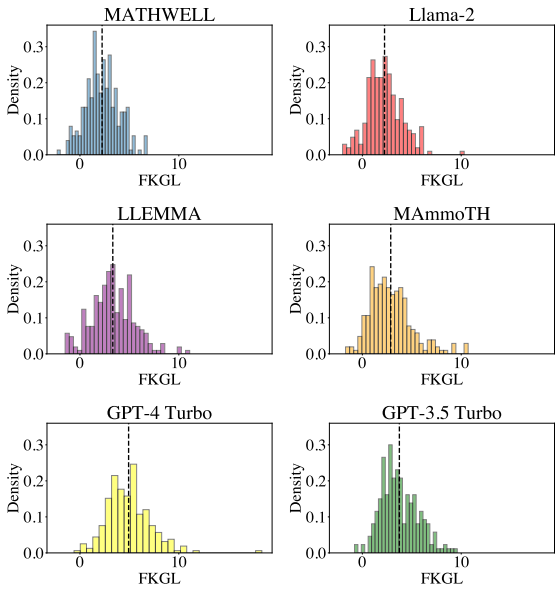

Figure 4: Flesch-Kincaid grade level (FKGL) distribution of model MWPs. Dotted lines show mean FKGL.

Dataset	PPL ↓	BERTScore F1
GSM8K	4.05 (1.18)	84.5
EGSM	2.44 (0.439)	84.3

Table 6: Automatic evaluation for datasets. PPL is perplexity and BERTScore F1 compares each dataset's questions to GSM8K. Bold denotes the lowest PPL. Standard deviations, if applicable, are in parentheses.

Tables 6 and 7, across models, datasets and comparisons, BERTScores are similar. Taken together, these findings suggest our synthetic data are similar to human quality, so our MWPs will likely be similar to ones students normally encounter.

MATHWELL Outputs Longer MaC Questions than Existing Open-source Models Longer token length may signal more complex MWPs, as longer MWPs often contain more information and operations than shorter ones. Comparing the average length of all MWPs to the length of MaC MWPs can determine if MaC MWPs are shorter or simpler. As shown in Table 7, MATHWELL's MaC MWPs are the longest of the open-source models considered, suggesting its MaC problems may be more complex. As shown in Table 12, MATHWELL and GPT-4 are also the only models for which MaC length is within a token of the overall average, providing evidence that its MaC MWPs are no simpler than its average MWP. While GPT-4 outputs longer MWPs than MATHWELL

	Model	PPL ↓	BF1	GSM BF1	MaC Length
API	GPT-4 Turbo	2.50 (0.03)	85.4	84.6	66.8 (2.62)
	GPT-3.5 Turbo	2.64 (0.03)	85.6	84.6	49.9 (1.16)
Public	LLEMMA	3.82 (0.10)	84.3	84.0	50.9 (2.89)
	MAmmoTH	2.76 (0.03)	86.0	84.6	44.4 (1.15)
	Llama-2	2.52 (0.03)	85.5	84.3	49.8 (1.19)
	MATHWELL	**2.44 (0.03)**	85.5	84.2	**54.1 (0.97)**

Table 7: Automatic evaluation metrics for each model. PPL is perplexity, BF1 is BERTScore F1, GSM BF1 compares each model's questions to GSM8K, MaC is meets all criteria, and Length is average token length. Bold indicates the lowest PPL and longest length for MaC generations for open-source models. Standard errors, where applicable, are in parentheses.

	Add.	Sub.	Mult.	Div.
Prompted for Operation (n=400)	**64.0**	**77.0**	**67.0**	**44.0**
Unprompted	6.07	10.12	6.07	1.62

Table 8: Percentage of questions generated that include only one operator when prompted to do so, compared to overall percentages that contain only that operator. Add., Sub., Mult., and Div. are addition, subtraction, multiplication, and division, respectively.

on average, MATHWELL's MWP length is still on par with those of the human-written grade school math datasets reported in Table 2. This finding further supports those discussed above showing that MATHWELL generates MWPs from a range of difficulties and complexities, which implies coverage of concepts throughout K-8 math education.

5 Controllability

Being able to control the math operations and topics present in MWPs generated by MATHWELL would maximize the model's educational applicability by allowing teachers to generate MWPs that target specific concepts they are teaching. To test MATHWELL's controllability, we prompted the model to generate questions involving only one of the operations from the list of Addition, Subtraction, Multiplication, and Division. Each prompt included 8-shot in-context examples and we generated 100 questions for each operation, totaling 400 generated questions. See Appendix A for the full prompt. We then measured the percent of generated questions that successfully include only the intended operator. We compare this to the distribution of questions produced by the unprompted model that only contain one of the operators.

As shown in Table 8, we find that requesting specific operators significantly increases the rate at which they are generated. There is room for improvement, especially for division, which we believe is clear future work. Regardless, this rate of operation controllability is competitive with the rate at which existing SOTA word problem generators successfully incorporate a pre-specified equation, with the exception of division controllability, which is similar to the rates of controllability reported in many of these works (Qin et al., 2023, 2024; Wang et al., 2021; Wu et al., 2022; Zhou and Huang, 2019; Zhou et al., 2023). These results paired with the high rate of topic controllability reported in Table 3 demonstrate that our model is non-trivially controllable.

6 Conclusions

We explore educational reference-free K-8 MWP generation and create the first dataset to train models for this task and the only one with teacher annotations. We demonstrate the quality of this dataset by using it to finetune MATHWELL, the first reference-free educational MWP generator. Our evaluations show that MATHWELL outperforms other open-source LLMs at reference-free MWP generation, matching the performance of GPT-4 while generating questions with a more appropriate reading level and that do not contain harmful inappropriate content. Further, we show that our dataset, EGSM, is comparable to existing math QA data while containing questions that are more educationally appropriate. These findings suggest that reference-free MWP generation is a feasible and practical alternative to traditional reference-dependent generators. Future research should train reference-free MWP generators to create questions aligned with specific grade levels and develop automated classification methods to reduce human annotation costs.

Limitations

One limitation of MATHWELL is that it is not specifically designed to generate questions aligned with pre-specified grade levels, which we chose to leave to future research due to the high cost of annotating questions for these characteristics. However, we demonstrate that our model is non-trivially promptable for specific math operations in Section 5, which highlights that exploring additional controllability is a promising area for future research. Additionally, MATHWELL is trained and evaluated for generating MWPs/solutions for K-8 stu-

dents only; therefore, we do not recommend using it to generate question/answer pairs for other grade levels or for other school subjects, which we believe are compelling areas for future work. Another limitation of MATHWELL's MWPs is that they are text-only, and students often encounter MWPs that are multi-modal (containing both images/tables/figures and text) in addition to those that are text-only. Generating multi-modal problems is thus a challenging and interesting area for future work. While we use a standard prompt for our experiments to simplify the evaluation framework and make fair comparisons across models, future work could conduct prompt tuning experiments to further improve model performance.

The high cost of human evaluation to ensure educational outputs is another limitation of automatic MWP generation broadly. In Appendix C.6, we explore training text classifiers to automatically score MATHWELL outputs. We show existing models can learn some features important for automatic classification, but need refinement in order to correctly classify unsolvable, inaccurate, or inappropriate questions. We hope these results and our large annotated dataset motivate future research in automatic classification efforts.

Another limitation of this work is the subjective nature of the appropriateness criteria. While it is critical model-generated questions are appropriate for students, it is hard to fully define all aspects of appropriateness and individuals may have differing opinions on the degree to which a question is appropriate or not. We chose to define several common reasons questions may not be appropriate for students (see Figure 8) and use teachers as annotators, but future research should continue to define this criteria and include multiple evaluators.

Ethics Statement

All data used to train MATHWELL come from open-access datasets and, therefore, should not contain any private sensitive information. MATHWELL may generate questions that are inappropriate for use in educational contexts and additional research should be conducted on the model before deploying it in classroom settings. Specifically, future research should continue to improve performance of text classifiers to filter out questions which are not appropriate for students.

Acknowledgements

We thank Zooniverse (Zooniverse) for providing a free and user-friendly platform for data annotation. We also thank our expert volunteer annotators for providing high-quality labels and feedback on our evaluation criteria/directions. Finally, we thank the University of Virginia's Research Computing team for maintaining excellent high-performance computing resources that allowed us to conduct this research.

References

Shivam Bansal Aggarwal, Chaitanya. textstat: Calculate statistical features from text.

Zhangir Azerbayev, Hailey Schoelkopf, Keiran Paster, Marco Dos Santos, Stephen McAleer, Albert Q. Jiang, Jia Deng, Stella Biderman, and Sean Welleck. 2023. Llemma: An Open Language Model For Mathematics. ArXiv:2310.10631 [cs].

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. 2022. Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback. ArXiv:2204.05862 [cs].

Matthew L. Bernacki and Candace Walkington. 2018. The role of situational interest in personalized learning. *Journal of Educational Psychology*, 110(6):864–881. Place: US Publisher: American Psychological Association.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training Verifiers to Solve Math Word Problems. ArXiv:2110.14168 [cs].

Gabriella Daroczy, Magdalena Wolska, Walt Detmar Meurers, and Hans-Christoph Nuerk. 2015. Word problems: a review of linguistic and numerical factors contributing to their difficulty. *Frontiers in Psychology*, 6:348.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient Finetuning of Quantized LLMs. ArXiv:2305.14314 [cs].

R. Flesch. 1948. A new readability yardstick. *The Journal of Applied Psychology*, 32(3):221–233.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. PAL: Program-aided Language Models. ArXiv:2211.10435 [cs].

Hongyi Guo, Yuanshun Yao, Wei Shen, Jiaheng Wei, Xiaoying Zhang, Zhaoran Wang, and Yang Liu. 2024. Human-instruction-free llm self-alignment with limited samples.

Ying Jiao, Kumar Shridhar, Peng Cui, Wangchunshu Zhou, and Mrinmaya Sachan. 2023. Automatic Educational Question Generation with Difficulty Level Controls. In *Artificial Intelligence in Education*, Lecture Notes in Computer Science, pages 476–488, Cham. Springer Nature Switzerland.

Jiwoo Kim, Youngbin Kim, Ilwoong Baek, JinYeong Bak, and Jongwuk Lee. 2023. It ain't over: A multi-aspect diverse math word problem dataset. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14984–15011, Singapore. Association for Computational Linguistics.

J. P. Kincaid, Jr. Fishburne, Rogers Robert P., Chissom Richard L., and Brad S. 1975. Derivation of New Readability Formulas (Automated Readability Index, Fog Count and Flesch Reading Ease Formula) for Navy Enlisted Personnel:. Technical report, Defense Technical Information Center, Fort Belvoir, VA.

Rik Koncel-Kedziorski, Ioannis Konstas, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2016. A theme-rewriting approach for generating algebra word problems. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1617–1628, Austin, Texas. Association for Computational Linguistics.

Shen-yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2020. A Diverse Corpus for Evaluating and Developing English Math Word Problem Solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 975–984, Online. Association for Computational Linguistics.

Swaroop Mishra, Matthew Finlayson, Pan Lu, Leonard Tang, Sean Welleck, Chitta Baral, Tanmay Rajpurohit, Oyvind Tafjord, Ashish Sabharwal, Peter Clark, and Ashwin Kalyan. 2023. Lila: A Unified Benchmark for Mathematical Reasoning. ArXiv:2210.17517 [cs].

Swaroop Mishra, Arindam Mitra, Neeraj Varshney, Bhavdeep Sachdeva, Peter Clark, Chitta Baral, and Ashwin Kalyan. 2022. NumGLUE: A Suite of Fundamental yet Challenging Mathematical Reasoning Tasks. ArXiv:2204.05660 [cs].

Arindam Mitra, Hamed Khanpour, Corby Rosset, and Ahmed Awadallah. 2024. Orca-math: Unlocking the potential of slms in grade school math.

Kashyapa Niyarepola, Dineth Athapaththu, Savindu Ekanayake, and Surangika Ranathunga. 2022. Math Word Problem Generation with Multilingual Language Models. In *Proceedings of the 15th International Conference on Natural Language Generation*, pages 144–155, Waterville, Maine, USA and virtual meeting. Association for Computational Linguistics.

Kole Norberg, Husni Almoubayyed, Stephen E Fancsali, Logan De Ley, Kyle Weldon, April Murphy, and Steve Ritter. 2023. Rewriting math word problems with large language models. *Proceedings of the Workshop on Empowering Education with LLMs-the Next-Gen Interface and Content Generation 2023 co-located with 24th International Conference on Artificial Intelligence in Education (AIED 2023)*, 3487:163–172.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer

McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. Gpt-4 technical report.

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. ArXiv:2203.02155 [cs].

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP Models really able to Solve Simple Math Word Problems? ArXiv:2103.07191 [cs].

Daniel Pearce, Faye Bruun, Kim Skinner, and Claricia Lopez-Mohler. 2013. What teachers say about student difficulties solving mathematical word problems in grades 2-5. *International Electronic Journal of Mathematics Education*, 8:3–19.

Longhu Qin, Jiayu Liu, Zhenya Huang, Kai Zhang, Qi Liu, Binbin Jin, and Enhong Chen. 2023. A Mathematical Word Problem Generator with Structure Planning and Knowledge Enhancement. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '23, pages 1750–1754, New York, NY, USA. Association for Computing Machinery.

Wei Qin, Xiaowei Wang, Zhenzhen Hu, Lei Wang, Yunshi Lan, and Richang Hong. 2024. Math Word Problem Generation via Disentangled Memory Retrieval. *ACM Transactions on Knowledge Discovery from Data*. Just Accepted.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108.

Sarah Schwartz. 2023. Why Word Problems Are Such a Struggle for Students—And What Teachers Can Do. *Education Week*.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H. Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 31210–31227. PMLR.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. 2022. Learning to summarize from human feedback. ArXiv:2009.01325 [cs].

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Shubham Toshniwal, Ivan Moshkov, Sean Narenthiran, Daria Gitman, Fei Jia, and Igor Gitman. 2024. Openmathinstruct-1: A 1.8 million math instruction tuning dataset. *arXiv preprint arXiv: Arxiv-2402.10176*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu,

Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Lieven Verschaffel, Stanislaw Schukajlow, Jon Star, and Wim Van Dooren. 2020. Word Problems in Mathematics Education: A Survey. *ZDM: The International Journal on Mathematics Education*, 52(1):1–16. Publisher: Springer ERIC Number: EJ1243930.

Candace Walkington and Matthew L. Bernacki. 2019. Personalizing Algebra to Students' Individual Interests in an Intelligent Tutoring System: Moderators of Impact. *International Journal of Artificial Intelligence in Education*, 29(1):58–88.

Candace Walkington, Virginia Clinton, and Pooja Shivraj. 2018. How readability factors are differentially associated with performance for students of different backgrounds when solving mathematics word problems. *American Educational Research Journal*, 55(2):362–414. Place: US Publisher: Sage Publications.

Candace A. Walkington. 2013. Using adaptive learning technologies to personalize instruction to student interests: The impact of relevant contexts on performance and learning outcomes. *Journal of Educational Psychology*, 105(4):932–945. Place: US Publisher: American Psychological Association.

Haoyu Wang, Guozheng Ma, Ziqiao Meng, Zeyu Qin, Li Shen, Zhong Zhang, Bingzhe Wu, Liu Liu, Yatao Bian, Tingyang Xu, Xueqian Wang, and Peilin Zhao. 2024. Step-on-feet tuning: Scaling self-alignment of llms via bootstrapping.

Zichao Wang, Andrew S. Lan, and Richard G. Baraniuk. 2021. Math Word Problem Generation with Mathematical Consistency and Problem Context Constraints. ArXiv:2109.04546 [cs].

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. ArXiv:2201.11903 [cs].

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi 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 M. Rush. 2020. HuggingFace's Transformers: State-of-the-art Natural Language Processing. ArXiv:1910.03771 [cs].

Qinzhuo Wu, Qi Zhang, and Xuanjing Huang. 2022. Automatic Math Word Problem Generation With Topic-Expression Co-Attention Mechanism and Reinforcement Learning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:1061–1072.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. ArXiv:2305.10601 [cs].

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.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. 2023. Scaling Relationship on Learning Mathematical Reasoning with Large Language Models. ArXiv:2308.01825 [cs].

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. 2023. MAmmoTH: Building Math Generalist Models through Hybrid Instruction Tuning. ArXiv:2309.05653 [cs].

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

Shen Zheng, Yuyu Zhang, Yijie Zhu, Chenguang Xi, Pengyang Gao, Xun Zhou, and Kevin Chen-Chuan Chang. 2023. GPT-Fathom: Benchmarking Large Language Models to Decipher the Evolutionary Path towards GPT-4 and Beyond. ArXiv:2309.16583 [cs].

Qingyu Zhou and Danqing Huang. 2019. Towards Generating Math Word Problems from Equations and Topics. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 494–503, Tokyo, Japan. Association for Computational Linguistics.

Zihao Zhou, Maizhen Ning, Qiufeng Wang, Jie Yao, Wei Wang, Xiaowei Huang, and Kaizhu Huang. 2023. Learning by analogy: Diverse questions generation in math word problem.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2020. Fine-Tuning Language Models from Human Preferences. ArXiv:1909.08593 [cs, stat].

Mingyu Zong and Bhaskar Krishnamachari. 2023. Solving Math Word Problems concerning Systems of Equations with GPT-3. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(13):15972–15979. Number: 13.

Zooniverse. Zooniverse.

A Prompting Process

A.1 Standard Prompts and Sampling Process

For all generations reported in this paper, we set temperature to 1.0 to strike a balance between creativity and encouraging the model to select the most

probable next token to guide effective solution generation. We also set the sampling parameter equal to true to further diversify outputs while also ensuring the most probable next token still receives the most weight. The GitHub repo associated with this paper has a sample generation script that uses this sampling approach. The exact prompts used in each experiment are reported below.

Prompt for Generating Synthetic Training Data Our standard prompt for evaluating the educational appropriateness of existing datasets (Section 3.1) and generating synthetic training data after the first stage of finetuning on public data (Section 3.2) is, "Write a grade school math word problem and Python function with a commented out step-by-step solution to solve the word problem." We begin each prompt with 8-shot examples, using randomly selected examples from MathInstruct GSM8K for evaluating existing datasets and a consistent set of 8 educationally appropriate MathInstruct GSM8K examples for generating synthetic training data.

Prompt for Interacting with MATHWELL and Comparing Models Our standard prompt for interacting with the MATHWELL model and for all experiments reported in Section 4 is, "Write a grade school math word problem about {topic} and Python function with a commented out step-by-step solution to solve the word problem."

In this prompt, topic is an optional argument, which we randomly select from the list of topics discussed in Appendix F. We begin every prompt with a random selection of 8-shot examples from EGSM.

Prompt for Controllability Experiment: Our prompt for the controllability experiment reported in Section 5 is: "Write a grade school math {operation} word problem about {topic} and Python function with a commented out step-by-step solution to solve the word problem. The question you write should only require {operation} to solve, meaning the solution should rely only on use of the {operator} operator."

In this prompt, operation is filled in with addition, subtraction, multiplication, or division, topic is randomly selected from the list of topics in Appendix F, and operator is filled in with +, -, *, or /. We begin every prompt with a random selection of 8-shot examples from EGSM that contain only the desired operation.

A.2 Expected Output Format

Using the prompting processes described above, the prompted model should output a question/answer pair with the solution being calculated by a Python function. The response should separate the question and solution using "Question:" and "Solution:," respectively. Below is a sample output from MATHWELL showing this output structure.

Sample Generation:
Question: Superman can fly 1200 miles per hour. He can also run 500 miles per hour. If he flies for 2 hours and runs for 3 hours, how many miles has he covered?

Solution:

```
def solution():
    #Superman can fly 1200 miles per hour
    superman_flight_speed = 1200
    #He can run 500 miles per hour
    superman_run_speed = 500
    #He flies for 2 hours
    flight_hours = 2
    #He runs for 3 hours
    run_hours = 3
    #The answer is
    result = (superman_flight_speed * flight_hours)
      + (superman_run_speed * run_hours)
    return result
```

A.3 Additional Suggested Prompting Strategies

We find MATHWELL is more likely to generate executable code when given a topic than when a topic is not specified. For example, when prompting our finetuned Llama-2 model before further training it on the EGSM data, we found the model generated executable code 63.1% of the time when given a topic, and only 32.3% of the time when a topic was not specified. As a result, for evaluating models in this paper, we provide them with a randomly selected topic, which also gives us the ability to assess their ability to effectively generate topic-specific word problems. Additionally, this evaluation strategy is aligned with how a teacher or student would use the model in practice, as they would want the generated questions to align with a particular topic. Qualitative evaluations of model generations also revealed that MATHWELL is more likely to generate executable code when the topic is more specific. For example, if their desired topic is superheroes, a

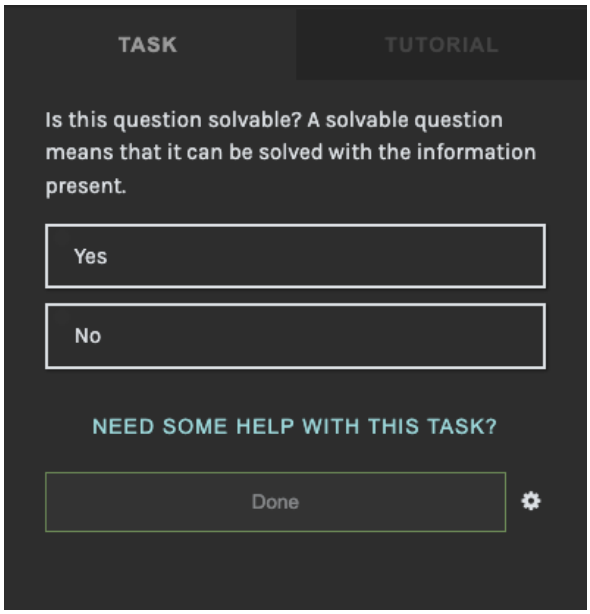

Figure 5: Solvability directions.

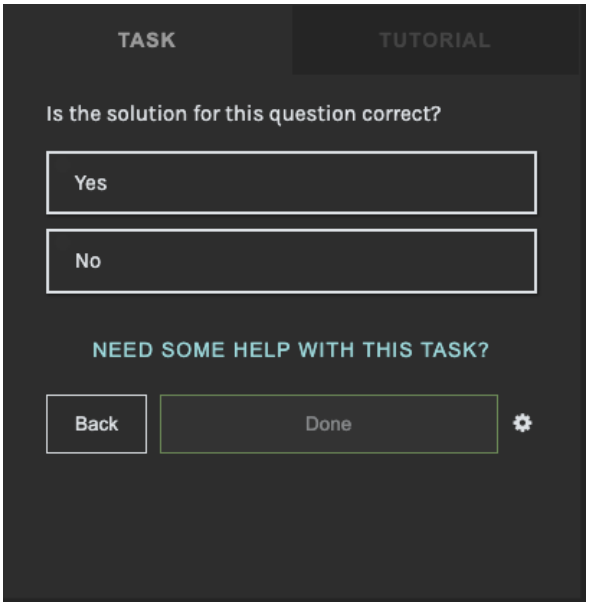

Figure 6: Accuracy directions.

user would have a higher likelihood of receiving a generation with executable code by prompting with a specific superhero (e.g., Superman) than leaving the topic general (e.g., superheroes).

B Annotation Process and Details

B.1 Annotators

All annotators had K-12 teaching experience or training, including a research team member who annotated every question. We had three primary annotators who reviewed at least 200 questions each in addition to our research team member. Our four primary annotators were seasoned educators, with an average of 5.75 years of education experience. They spent an average of 48 seconds per question, totaling 96.3 hours of expert annotation throughout our full human evaluation of over 5,000 model-generated word problems.

B.2 Validating Final Evaluation Labels

For annotating the 250 samples from each model for our experiments reported in Section 4, we randomized questions from each model and had them blindly reviewed by one of our highly trained annotators with K-12 teaching experience. To evaluate the quality of these labels, we had 357 randomly reviewed by one additional annotator and 60 randomly reviewed by two additional annotators. The annotators agreed on solvability 89.3 ± 3.0% of the time, accuracy 95.5 ± 2.0% of the time, appropriateness 80.4 ± 3.8% of the time, all three labels 67.8 ± 4.5% of the time, and MaC 78.3 ± 3.9% of the time. The agreement rates for accuracy and solvability are higher than reported in recent studies that explore human alignment of LLM outputs, and the agreement rates for appropriateness and MaC are on par with these studies (Bai et al., 2022; Ouyang et al., 2022; Stiennon et al., 2022; Ziegler et al., 2020). Additional analysis reveals that most annotator disagreement (80.1%) was due to the primary annotator being more conservative than the additional annotators by labeling questions as not having the desired criteria when the additional annotators rated them as having the desired criteria. As a result, we chose to use the labels from the primary annotator when reporting final results to be conservative, though we also found the results do not vary when switching labels based on annotator disagreement. Annotators were least likely to disagree on labels for MATHWELL and GPT-4 Turbo outputs and our primary annotator was not more likely to rate MATHWELL outputs as having the desired criteria than the additional annotators. Taken together, this evidence suggests our final labels are highly accurate.

B.3 Defining Appropriateness

Teachers participated in designing our appropriateness metric. Our research team includes a former K-12 teacher, who led metric development. We iteratively refined the metric with feedback from teacher annotators using sample model outputs. This process led to the definition described in the paper

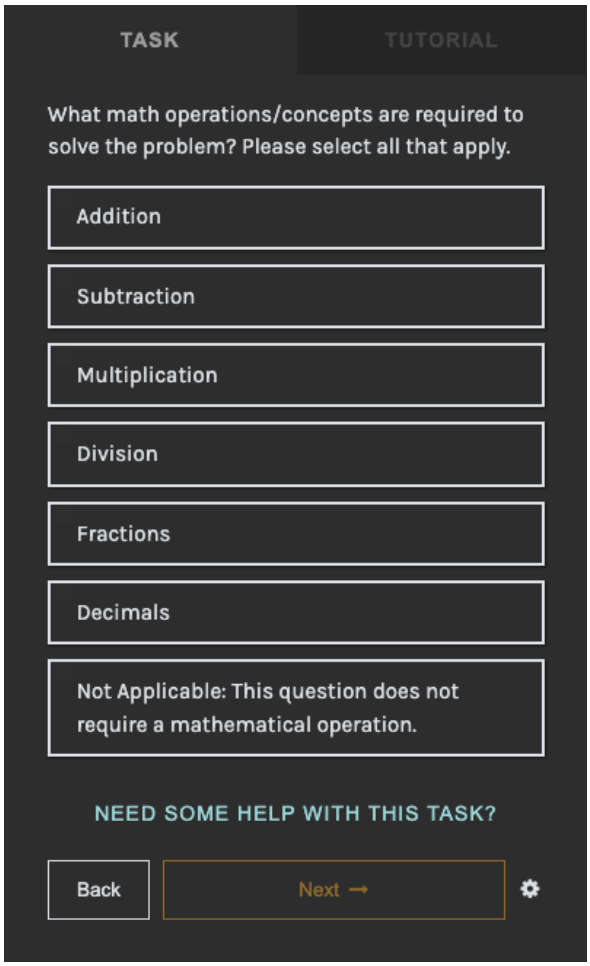

Figure 7: Labeling operations directions.

along with the annotator directions and response options shown in Figure 8.

B.4 Annotation Interface

We used Zooniverse (Zooniverse) to collect our human annotation data. Figures 5, 6, 7 and 8 show the instructions each annotator was given for each of our evaluation criteria.

B.5 Annotated Data Characteristics

Our full annotated dataset consists of 5,084 question/answer pairs with teacher annotations for solvability, accuracy, appropriateness, and MaC. The data are comprised of the 3,234 word problem dataset used to generate training data for MATHWELL in addition to the 250 evaluation set for each model described in Section 4, the 100 questions we evaluated in Section 3.1, and 250 questions we evaluated in Appendix C.1 from training our model using the 2nd finetuning stage only. Based on our annotations, 82.2% of the question/answer pairs are solvable, 87.3% have accurate solutions, 78.1% are appropriate, and 58.4% meet all criteria.

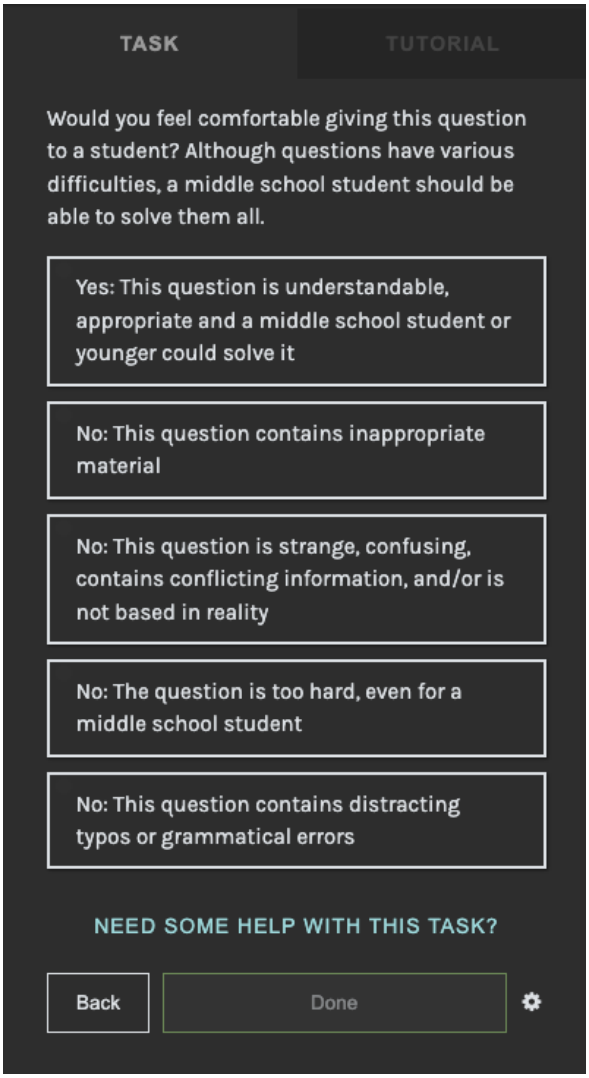

Figure 8: Appropriateness directions.

C Additional Experiments

C.1 Finetuning Ablation

As shown in Table 9, MATHWELL shows a statistically and/or substantively meaningful improvement in all metrics after the second stage of finetuning on high-quality synthetic data labeled by domain experts relative to just doing either finetuning stage alone. These results lead us to conclude that the second stage of finetuning is critical for improving the model's ability to generate questions that are educationally appropriate, or MaC.

C.2 Logistic Regression for Predicting MaC

As shown in Table 10, the coefficients for all models except GPT-4 for MaC remain negative and statistically significant relative to MATHWELL

Model	Solv.	Acc.	App.	MaC	EC	EC/MaC
MATHWELL (1st Stage Only)	82.3 (0.67)	85.8 (0.66)	80.1 (0.77)	58.9 (0.87)	63.1 (0.73)	37.2 (0.55)
MATHWELL (2nd Stage Only)	86.0 (2.20)	91.6 (1.89)	76.7 (2.89)	60.8 (3.09)	50.9 (1.20)	31.0 (1.58)
MATHWELL (1st and 2nd Stage)	**89.2 (1.97)**	**96.9* (1.17)**	**86.5* (2.29)**	**74.8** (2.75)**	**66.4** (1.00)**	**49.6** (1.83)**

Table 9: Average metrics for MATHWELL after the first and second round of finetuning compared to the model finetuned with both stages. Solv., Acc., App., MaC, and EC/MaC are solvability, accuracy, appropriateness, meets all criteria, and the estimated share of questions that MaC and have executable code, respectively. Bold indicates the best performance in each metric and a * or ** indicates the difference between the best performance and second best performance is statistically significant at the p<.05 or p<.01 level, respectively. Standard errors are in parentheses.

Predictor	Coefficient	SE	Z	p
Constant	1.648	0.182	9.053	0.000**
GPT-4 Turbo	-0.053	0.251	-0.212	0.832
GPT-3.5 Turbo	-0.735	0.235	-3.121	0.002**
LLEMMA	-2.441	0.267	-9.138	0.000**
MAmmoTH	-1.009	0.231	-4.363	0.000**
Llama-2	-0.587	0.241	-2.435	0.015*
Constant	1.496	0.254	5.895	0.000**
GPT-4 Turbo	0.064	0.262	0.243	0.808
GPT-3.5 Turbo	-0.496	0.247	-2.008	0.045*
LLEMMA	-2.279	0.276	-8.269	0.000**
MAmmoTH	-0.867	0.238	-3.638	0.000**
Llama-2	-0.532	0.244	-2.183	0.029*
Addition	0.130	0.153	0.850	0.395
Subtraction	0.234	0.165	1.413	0.158
Multiplication	-0.123	0.165	-0.748	0.454
Division	-0.124	0.193	-0.645	0.519
Fractions	-0.152	0.323	-0.471	0.638
Decimals	-0.345	0.208	-1.658	0.097

Table 10: Logistic regression results for meets all criteria (MaC), with and without controlling for the impact of question type. These results only consider questions which are labeled as solvable. The reference model for the constant is MATHWELL. A * or ** indicates statistical significance at the p<0.05 or p<0.01 level, respectively.

when controlling for the type of mathematical operation. This finding supports the assertion that MATHWELL is more capable than these models of generating MaC questions regardless of the operation considered, even if it is less likely to generate questions from the more complex mathematical operations. Table 10 also shows that the difference in MaC performance between MATHWELL and GPT-4 remains statistically insignificant even when controlling for the impact of operations, highlighting that both models perform similarly on this task.

C.3 Accuracy by Question Type

As shown in Table 11, MATHWELL's accuracy does not differ significantly or substantively by operation, as its accuracy remains above 90% for all operations, while the other models except GPT-4 have a significant and/or substantive gap in their accuracy for the operation they perform best on relative to the operation they perform worst on. While MAmmoTH outperforms MATHWELL for addition, multiplication, and division, MATHWELL performs better in the other three operations and in overall accuracy. Further, MATHWELL outperforms GPT-4 in addition, subtraction, and decimals while maintaining similar performance in the other three operations.

	Model	Add.	Sub.	Mult.	Div.	Frac.	Dec.
API	GPT-4 Turbo	96.0	94.3	95.9	93.3	100.0	90.9
	GPT-3.5 Turbo	91.0	95.2	86.6	87.5	71.4	90.2
Public	LLEMMA	76.2	72.3	63.4	56.0	50.0	63.2
	MAmmoTH	**96.5**	96.3	**96.8**	**97.6**	87.5	91.3
	Llama-2	89.3	91.1	87.5	80.0	82.4	75.0
	MATHWELL	96.1	**97.4**	94.5	91.3	**100.0**	**94.1**

Table 11: Accuracy by operation. Add., Sub., Mult., Div., Frac., Dec., No Ops, Total Ops, and MaC are addition, subtraction, multiplication, division, fractions, decimals, no operations, total operations, and meets all criteria, respectively. Bold indicates the best open-source performance in each operation. A bold model name indicates the difference between that model's operation with the highest accuracy and lowest accuracy is statistically significant at the p<0.05 level.

C.4 FKGL Comparisons

As shown in Figure 10, the FKGL distribution for all versus MaC questions does not significantly vary for all models except for LLEMMA, whose MaC questions tend to have lower FKGL than its average question. Figure 9 compares the FKGL distribution of EGSM questions to the three other existing datasets that are mathematically appropriate for grade school students. Fewer of EGSM's questions than other existing datasets are written at a grade level beyond 8th grade.

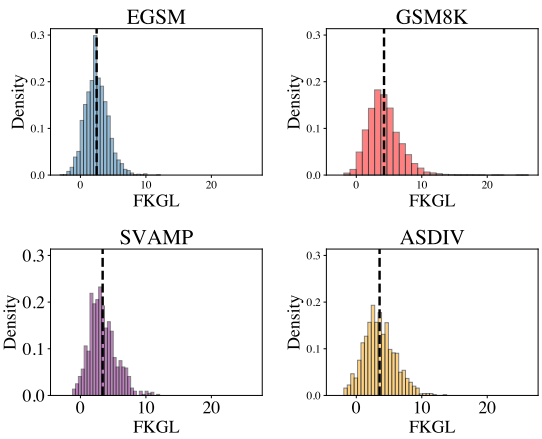

Figure 9: Flesch-Kincaid grade level (FKGL) distribution of training datasets. Dotted lines show the mean for each dataset.

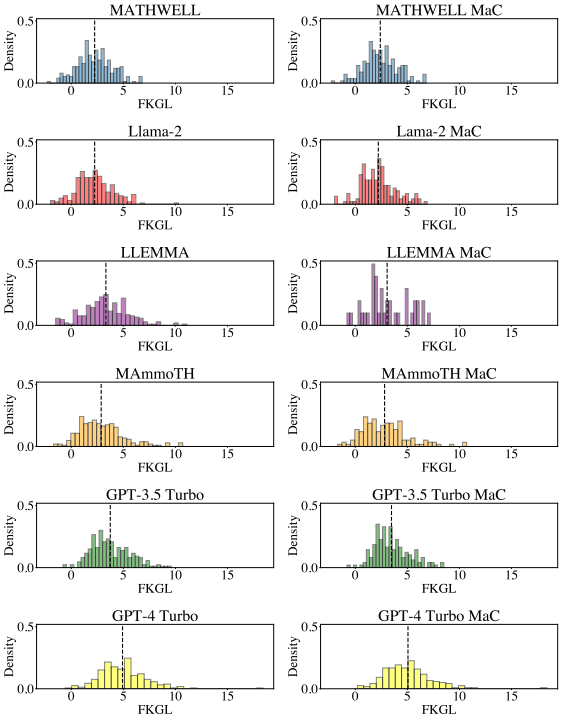

Figure 10: Flesch-Kincaid grade level (FKGL) distribution of model generations for all versus MaC questions. Dotted lines show the mean for each model.

C.5 Additional Automatic Evaluation Comparisons

As shown in Table 12, the findings from the PPL experiment remain unchanged when PPL is evaluated with GPT-2 (MATHWELL outputs still have the lowest PPL of all models considered). Additionally, PPL does not significantly vary for all models except for LLEMMA when comparing all to MaC generations. In the table, we also compare the BERTScore between all and MaC questions from each model and show they are similar. While LLEMMA has a longer average token length for all its generations than MATHWELL, MATHWELL's MaC average token length is the longest of the open-source models considered.

C.6 Exploring Automatic Classification

Given the high cost of human evaluation, we consider automatically evaluating model outputs by training DistilBERT (Sanh et al., 2019) classifiers for solvability, accuracy and appropriateness using our annotated data, excluding the evaluation samples from each model. We use an 80/10/10 train, validation and test split. The solvability dataset has 3,234 rows, the accuracy dataset has 2,830 rows, and the appropriateness dataset has 2,660 rows. We exclude unsolvable questions from the accuracy and appropriateness datasets to promote specialization, as these questions do not have a solution and are inappropriate. Each dataset is unbalanced, with 82.3% of questions labeled as solvable, 85.8% labeled as accurate, and 80.1% labeled as appropriate. As a result, we modify the training objective to weight the loss based on the inverse proportion of observations from each class. We train each model for 8 epochs using the HuggingFace library (Wolf et al., 2020) with strict regularization (weight decay = 0.9) and use the checkpoint with the lowest validation loss for testing.

As shown in Table 13, despite weighting the loss, each classifier has a lower balanced accuracy than its overall accuracy. The classifiers perform similarly on outputs from all models except LLEMMA, for which performance is significantly lower. As would be expected, performance for all models tends to be lower than on the test dataset. Excluding LLEMMA and the appropriateness classifier's precision on MAmmoTH outputs, each classifier has fairly high precision, recall and F1, suggesting they are effectively able to label solvable, accurate, and appropriate questions. The classifiers tend to have low ROC AUC across sources, suggesting they do not perform well at different prediction cutoffs. As a whole, Table 13 provides evidence that text classifiers can learn some features that are important for labelling MATHWELL outputs, but future research should use more balanced datasets to improve their ability to label unsolvable, inaccurate, and inappropriate questions.

	Model	GPT-2 PPL ↓	PPL ↓	MaC PPL ↓	BF1	MaC BF1	All/MaC BF1	GSM BF1	Length	MaC AL
API	GPT-4 Turbo	10.68 (2.90)	2.50 (0.03)	2.49 (0.03)	85.4	85.5	85.5	84.6	**66.0 (2.12)**	66.8 (2.62)
	GPT-3.5 Turbo	10.57 (3.60)	2.64 (0.03)	2.70 (0.04)	85.6	85.8	85.7	84.6	52.8 (1.00)	49.9 (1.16)
Public	LLEMMA	15.20 (8.40)	3.82 (0.10)	3.14 (0.10)	84.3	85.3	84.6	84.0	*56.4 (1.44)*	50.9 (2.89)
	MAmmoTH	11.96 (4.26)	2.76 (0.03)	2.74 (0.04)	86.0	86.4	86.1	84.6	45.9 (1.13)	44.4 (1.15)
	Llama-2	9.97 (4.12)	2.52 (0.03)	2.52 (0.04)	85.5	85.8	85.6	84.3	51.6 (0.99)	49.8 (1.19)
	MATHWELL (Ours)	**9.90 (3.84)**	**2.44 (0.03)**	**2.43 (0.03)**	85.5	85.7	85.6	84.2	54.7 (0.86)	**54.1 (0.97)**

Table 12: Automatic evaluation metrics for each model. PPL is perplexity (evaluated by Llama-2 70B unless otherwise noted), BF1 is BERTScore F1, MaC is meets all criteria (MaC), All/MaC BF1 compares all to MaC questions, GSM BF1 compares each model's questions to GSM8K, and Length is average token length. Bold indicates the lowest PPL and longest length for open-source models. Standard errors, where applicable, are in parentheses.

	Solvability						Accuracy						Appropriateness					
Source	**Acc.**	**BA**	**P**	**R**	**F1**	**RA**	**Acc.**	**BA**	**P**	**R**	**F1**	**RA**	**Acc.**	**BA**	**P**	**R**	**F1**	**RA**
Annotated Test	78.9	61.9	86.7	88.0	87.4	0.701	81.6	53.8	87.2	92.2	89.6	0.649	82.7	72.0	89.8	88.9	89.4	0.793
GPT-4 Turbo	83.6	62.3	96.2	86.1	90.9	0.560	95.8	54.8	96.2	99.6	97.8	0.637	77.2	59.0	87.2	85.5	86.4	0.579
GPT-3.5 Turbo	81.9	52.3	88.5	91.3	89.9	0.519	88.6	55.2	90.5	97.4	93.9	0.541	78.1	64.3	82.1	91.0	86.3	0.696
LLEMMA	48.8	49.8	48.7	90.2	63.2	0.542	59.0	47.1	62.5	89.7	73.7	0.492	45.1	50.1	41.8	80.4	55.0	0.489
MAmmoTH	84.4	52.5	87.4	95.9	91.4	0.700	93.5	57.9	95.7	97.6	96.6	0.669	66.8	54.2	69.8	89.8	78.6	0.574
Llama-2	82.4	53.1	84.9	96.2	90.2	0.619	85.2	51.6	89.8	94.1	91.9	0.677	73.8	50.4	81.1	88.2	84.5	0.628
MATHWELL	85.2	54.3	90.1	93.7	91.9	0.599	94.6	62.7	97.7	96.8	97.2	0.790	77.6	63.1	90.4	82.9	86.5	0.669

Table 13: DistilBERT (Sanh et al., 2019) text classifier performance for solvability, accuracy, and appropriateness. Acc., BA, P, R, and RA are accuracy, balanced accuracy, precision, recall and ROC AUC, respectively.

C.7 Important Criteria for Reference-free Word Problem Generator Training Data

In addition to high-quality grammar and problems that are similar to those students encounter in the classroom, the characteristics we find most important for training reference-free word problem generators are PoT solutions written as Python functions and questions that are educationally appropriate for K-8 students. Regarding the former, when we modified our prompt to ask for a Python function solution instead of a Python code solution, the share of question/answer pairs with executable code from an early version of MATHWELL increased from 18.9% to 29.0%. For educational appropriateness, when we used the GSM-Hard (Gao et al., 2023) dataset as part of MATHWELL's training data, the model often generated questions with large numbers that are inappropriate for K-8 students. We further show that existing data are not educationally appropriate in Section 3.1. As shown in Table 2, EGSM is the only math QA dataset that has these two characteristics.

C.8 Early MATHWELL Experimentation

In addition to training reference-free question/answer pair generators, we also experiment with training reference-free question generation models. Our theory is that if we could train a model to generate questions effectively, we could pass those questions to a math QA model to retrieve answers automatically. To test this theory, we finetune both Llama-2 and MAmmoTH as question generators using the same training data discussed in Appendix D.1, except for excluding the solution for each question and modifying the standard prompt to ask the model to generate a question only. We then sample and evaluate 100 generations from each model. We find that MAmmoTH performs better than Llama-2 at this task, but neither model performs optimally. For example, only 19% of the MAmmoTH generations include the requested topic and 52.6% are solvable. Therefore, based on the results we report in Table 3, we conclude that it is more efficient to train a reference-free question/answer pair generator than question generator.

D Finetuning Details

D.1 Initial Finetuning

We re-format MathInstruct GSM8K (Yue et al., 2023) into an Alpaca-style (Taori et al., 2023) instruction dataset of question/answer pairs with a standard prompt asking the model to generate a grade school math word problem. We also include MathInstruct MATH and TheoremQA PoT (Yue et al., 2023) question/answer pairs with a standard prompt asking the model to generate a challenge

math problem to expose the model to more complex code and further promote mathematical reasoning. We then finetune Llama-2 (70B) on this dataset of 25,926 rows using QLoRA (Dettmers et al., 2023) for 4,250 steps. **Our finetuning process uses the following training resources and hyperparameters:** two A100 GPUs, a learning rate of 1e-4, LoRA modules at every model layer, per-device batch size of 1, and a 3% warm-up ratio. The training script in the GitHub repo linked to this paper contains a full list of hyperparameters.

D.2 Secondary Finetuning on Expert-annotated Data

Inspired by recent works that iteratively finetune LLMs (Guo et al., 2024; Wang et al., 2024), including OpenAI's FeedME approach (Zheng et al., 2023), we next aim to evaluate and improve how well MATHWELL meets our three main criteria: solvability, accuracy, and appropriateness. However, these properties are unannotated in prior datasets and we find they are ill-suited for training educational generators in Section 3.1. Therefore, we few-shot prompt the finetuned model and generate 3,234 problems. We then acquire annotations for solvability, accuracy, and appropriateness from teachers. Further details about the annotation process and annotator agreement are in Section 3.2 and Appendix B. These annotations identified 1,906 problems that are simultaneously solvable, accurate, and appropriate, or meet all criteria (MaC). We then finetune MATHWELL on these MaC outputs for 1,250 more steps using the same hyperameters discussed above.

E Annotated Example Generations

The sections below display randomly sampled generations from each model, including examples of both questions that do not meet evaluation criteria and those that do. For the examples that do not meet the specified criteria, we provide a short natural language annotation explaining the error.

E.1 GPT-4 Turbo

E.1.1 Solvability

Positive Example In a volleyball match, Team A scored twice as many points as Team B. If Team B scored 15 points, how many points in total were scored in the match?

Negative Example In a Fortnite Battle Royale match, there are 100 players. If one player can eliminate 4 opponents, how many players in the game can achieve such an elimination count, assuming no other players are eliminated?

Negative Example Reasoning It is not entirely clear what this question is asking. If it defined how many players should be left in the game after eliminations, it would be possible to determine how many players can be eliminated but, without defining that, it is not clear what the student needs to do to solve this problem.

E.1.2 Accuracy

Positive Example Jenny has 15 cats. She buys 2 new cat towers, each can hold 4 cats. If she places as many cats as possible on the cat towers, how many cats are not on the towers?

```
Solution:
def solution():
    # Jenny starts with 15 cats
    total_cats = 15
    # Each new cat tower holds 4 cats
    tower_capacity = 4
    # She buys 2 new cat towers
    total_towers = 2
    # The total number of cats the towers can hold is
    total_capacity = tower_capacity * total_towers
    # The number of cats not on the towers is
    cats_not_on_towers = total_cats - total_capacity
    return cats_not_on_towers

# Uncomment the line below to test the function
with the given values
# print(solution())
```
[4]

Negative Example Mrs. Johnson has 12 adult cats and each adult cat has 3 kittens. After several months, those kittens grow up and each new adult cat also has 2 kittens. How many kittens are there in total now?

```
Solution:
def solution():
    # Mrs. Johnson has 12 adult cats originally
    adult_cats_initial = 12
    # Each adult cat has 3 kittens
    kittens_per_adult_initial = 3
```

[4] As discussed in Appendix G, GPT-4 tends to output additional information beyond what is requested in the prompt and shown in few-shot examples, such as example usage or comments about the Python function solution, as is included here.

```
    # Calculate the initial number of kittens
    initial_kittens = adult_cats_initial *
        kittens_per_adult_initial
    # The initial kittens grow up to become adults
    new_adult_cats = initial_kittens
    # Each new adult cat has 2 kittens
    kittens_per_new_adult = 2
    # Calculate the new kittens born from the
    # grown-up kittens
    new_kittens = new_adult_cats *
        kittens_per_new_adult
    # The total number of kittens now is the sum
    # of the initial kittens and new kittens
    total_kittens = initial_kittens + new_kittens
    return total_kittens

# Call the function and print the result
print(f"There are {solution()} kittens in total
now.")
```

Negative Example Reasoning The solution adds the original number of kittens to the number of new kittens, which is incorrect because the original kittens are now considered adults based on the text of the question.

E.1.3 Appropriateness

Positive Example During a week of intense battles, the Power Rangers defeated 15 evil monsters on Monday, twice as many on Tuesday as on Monday, and 10 fewer on Wednesday than on Tuesday. How many monsters did they defeat in total from Monday to Wednesday?

Negative Example Serena Williams has won a certain number of tennis matches this season. If she wins 8 more matches, she will double the number she has won so far. How many matches has Serena won this season?

Negative Example Reasoning This question does not require any mathematical operation to solve and is therefore not appropriate. The question tells you Serena's number of wins will double if she wins 8 more matches, which means she had to have won 8 matches so far.

E.2 GPT-3.5 Turbo

E.2.1 Solvability

Positive Example During a basketball game, the Lakers made a total of 84 points. LeBron James scored 35 points, and Anthony Davis scored 20 points. How many points were scored by the rest of the Lakers team?

Negative Example The Ninja Turtles went to the pizza parlor and ordered 8 pizzas. Each pizza had 8 slices. If they each ate 2 slices, how many slices of pizza were left over?

Negative Example Reasoning This question does not define the number of Ninja Turtles. While many students might know there are typically 4 main Ninja Turtles, not every student would know this and all math questions should still define the key variables you need to answer them.

E.2.2 Accuracy

Positive Example Black Panther has 1500 vibranium shards. He needs to divide them equally amongst his 5 closest allies. How many vibranium shards will each ally receive?

Solution:

```
def solution():
    #Black Panther has 1500 vibranium shards
    vibranium_total = 1500
    #He needs to divide them equally amongst his 5
    #allies
    allies = 5
    #The answer is
    result = vibranium_total / allies
    return result
```

Negative Example A dog shelter has 40 dogs. If each dog needs 2 cups of food per day, and a bag of food contains 40 cups, how many bags of food will the shelter need for a 30-day period?

Solution:

```
def solution():
    #The shelter has 40 dogs
    dogs = 40
    #Each dog needs 2 cups of food per day
    food_per_dog = 2
    #A bag of food contains 40 cups
    food_per_bag = 40
    #For 30 days, the shelter will need
    total_food_needed = dogs * food_per_dog * 30
    #The answer is
    result = total_food_needed // food_per_bag + 1
    return result
```

Negative Example Reasoning The solution adds an additional bag of food to the total assuming that the answer is a decimal and, therefore, that the

shelter would need to buy another full bag since a partial bag is not possible. However, since the answer is whole number, this additional bag leads to outputting an incorrect result.

E.2.3 Appropriateness

Positive Example A cheerleading team has 20 members. They want to evenly divide into 4 squads for a competition. How many members will be on each squad?

Negative Example In a Pokémon battle, Pikachu has a 60% chance of winning each round. If Pikachu and Ash battle for 5 rounds, what is the probability that Pikachu wins all 5 rounds?

Negative Example Reasoning This question is too hard for a middle school student. It assumes a student knows about the probability of independent events, which is typically not covered until high school or an introductory statistics college course.

E.3 LLEMMA

E.3.1 Solvability

Positive Example Steph Curry makes 2133 free throws for the year. He hits 342 more free throws than he misses. How many free throws does Steph Curry miss?

Negative Example There are 5175 Pokémon available to battle in Pokémon Sword and Shield. There are 310 ground-type Pokémon and 182 water-type Pokémon. There are 610 Pokémon that are not ground nor water type. How many Pokémon are ground-type or water-type?

Negative Example Reasoning The question contains conflicting information in that it defines the number of ground and water type Pokémon twice.

E.3.2 Accuracy

Positive Example Naruto eats 4 bananas a day. How many bananas does he eat in a month if there are 30 days in a month?

Solution:
```
def solution():
  #Naruto eats 4 bananas a day
  bananas_per_day = 4
  #There are 30 days in a month
  days_in_month = 30
  #The answer is
  result = bananas_per_day * days_in_month
  return result
```

Negative Example 12 cats eat 400 pounds of cat food every Saturday morning. 15 cats eat 500 pounds of cat food on Sunday morning. How many pounds of cat food are eaten in total?

Solution:
```
def solution():
  #12 cats eat 400 pounds of cat food every
  #Saturday morning
  cat_food_satuday_morning = 12*400
  #15 cats eat 500 pounds of cat food on Sunday
  #morning
  cat_food_sunday_morning = 15*500
  #In total,
  result = cat_food_satuday_morning +
    cat_food_sunday_morning
  return result
```

Negative Example Reasoning The model multiplies the number of cats by the amount of cat food eaten, rather than adding the two sums of cat food together.

E.3.3 Appropriateness

Positive Example Hello Kitty makes 18 bracelets in 4 hours. How many bracelets per hour does she make?

Negative Example Batman caught a baddie with his trademark punch. Each punch knocks out 7 baddies. If Batman has thrown 60 punches, how many baddies has he knocked out?

Negative Example Reasoning While this question is comical, it is not appropriate for a K-8 student because it involves physically harming another person.

E.4 MAmmoTH

E.4.1 Solvability

Positive Example The 5 Power Rangers, Red, Blue, Black, Yellow, and Pink, are eating pizza. Red ate 31 slices, Blue ate 28 slices, Black ate 23 slices, Yellow ate 21 slices, and Pink ate 15 slices. How many slices of pizza did they eat in total?

Negative Example In Fortnite, the player has 5 health points. When the player is hit by an enemy, they lose 2 health points. How many health points does the player have left?

Negative Example Reasoning This question defines how many health points a player loses when they get hit by an enemy, but it does not define how many times the player got hit, so it is impossible to determine how many health points they have left.

E.4.2 Accuracy

Positive Example Harry Potter and his friends have just finished their exams and are looking forward to a well-deserved break. They decide to go on a camping trip together. They have 120 Galleons between them. They spend 30 Galleons on food in the morning and 20 Galleons on food in the afternoon. They have 20 Galleons left. How many Galleons did they spend in the evening?

```
Solution:
def solution():
    # total amount of Galleons they had
    total_galleons = 120
    # amount spent on food in the morning
    morning_food = 30
    # amount spent on food in the afternoon
    afternoon_food = 20
    # amount left
    left_galleons = 20
    # calculate the amount spent in the evening
    evening_food = total_galleons - morning_food -
        afternoon_food - left_galleons
    return evening_food
print(solution())
```
[5]

Negative Example Captain Marvel has 100 friends on Facebook. She has 40 more friends than the average number of friends her friends have. How many friends does the average friend of Captain Marvel have?

```
Solution:
def solution():
    #Captain Marvel has 100 friends on Facebook
    captain_marvel_friends = 100
    #She has 40 more friends than the average
    #number of friends her friends have
    captain_marvel_friends_more = 40
    #The average number of friends her friends have
    average_friends = (captain_marvel_friends -
        captain_marvel_friends_more) / 2
    return average_friends
```

[5]MAmmoTH is prone to ending its generations with print(solution()) because its PoT training data always ends with a print statement (Yue et al., 2023).

Negative Example Reasoning The model calculates an average rather than subtract the average number of friends Captain Marvel's friends have from her number of friends.

E.4.3 Appropriateness

Positive Example Taylor Swift has 11 Grammys, 29 AMAs, 12 CMAs, 8 ACMs and 35 BMAs. How many awards has she won in total?

Negative Example The soccer team has 32 players. Each player has 2 legs. How many legs does the team have?

Negative Example Reasoning While this question is solvable, it is not appropriate because it is strange to ask how many legs a soccer team has.

E.5 Llama-2

E.5.1 Solvability

Positive Example The Pittsburgh Steelers have 11 players on the field. 3 of the players are from the 10th grade, 6 are from the 11th grade, and the rest are from the 12th grade. How many players are from the 12th grade?

Negative Example The cheerleaders at the school are planning a pep rally. They need to make 200 pom-poms. Each pom-pom requires 20 minutes to make. How many hours will it take to make 200 pom-poms?

Negative Example Reasoning The question does not define how many cheerleaders there are, so it is impossible to determine how long it will take them to make the pom-poms.

E.5.2 Accuracy

Positive Example Barbie has 100 pairs of shoes. She gave 20 pairs of shoes to Ken, 30 pairs of shoes to Skipper, and 40 pairs of shoes to Stacie. How many pairs of shoes does Barbie have left?

```
Solution:
def solution():
    #Barbie started with 100 pairs of shoes
    shoes_initial = 100
    #She gave 20 pairs of shoes to Ken
    shoes_ken = 20
    #She gave 30 pairs of shoes to Skipper
    shoes_skipper = 30
    #She gave 40 pairs of shoes to Stacie
    shoes_stacie = 40
    #The answer is
```

```
  result = shoes_initial - shoes_ken -
    shoes_skipper - shoes_stacie
  return result
```

Negative Example A basketball team scored 120 points in a game. The team scored 30 points in the first quarter, 35 points in the second quarter, 20 points in the third quarter, and 35 points in the fourth quarter. How many points did the team score in the second half of the game?

Solution:

```
def solution():
  #The team scored 120 points in a game
  total_points = 120
  #30 points in the first quarter
  points_first_quarter = 30
  #35 points in the second quarter
  points_second_quarter = 35
  #20 points in the third quarter
  points_third_quarter = 20
  #35 points in the fourth quarter
  points_fourth_quarter = 35
  #The answer is
  result = points_second_quarter +
    points_third_quarter + points_fourth_quarter
  return result
```

Negative Example Reasoning The model correctly defines the variables, but it incorrectly adds the points scored in the second quarter to the total for the points scored in the second half.

E.5.3 Appropriateness

Positive Example Patrick Mahomes is a professional football player. He is 6 feet and 3 inches tall. 1 inch is 2.54 centimeters. How many centimeters tall is Patrick Mahomes?

Negative Example A cat has 100 kittens. 20 of them are calico, 30 are tabby, and the rest are siamese. How many kittens are siamese?

Negative Example Reasoning This question is not based in reality, as it is not possible for one cat to birth 100 kittens, nor is it possible for them to be different breeds.

E.6 MATHWELL

E.6.1 Solvability

Positive Example A soccer team has 11 players on the field at a time. They have 22 more players on the bench. How many players are on the team in total?

Negative Example Steph Curry, the NBA star, is 6.2 feet tall. He wears shoes that are 1.5 inches thick. How tall is he in inches? (1 foot is 12 inches)

Negative Example Reasoning This question is not solvable because it is vague: we do not know if we are calculating his height in inches with or without shoes on.

E.6.2 Accuracy

Positive Example In the game, Super Mario can stomp on 30 Goombas, 20 Koopas, and 10 Piranha Plants. How many enemies does he stomp on total?

Solution:

```
def solution():
  #Super Mario can stomp on 30 Goombas
  goombas = 30
  #20 Koopas
  koopas = 20
  #and 10 Piranha Plants
  piranha_plants = 10
  #The answer is
  result = goombas + koopas + piranha_plants
  return result
```

Negative Example A Minecraft player has 100000 blocks. 20000 of the blocks are dirt, 30000 of the blocks are stone, 20000 of the blocks are wood, and 30000 of the blocks are diamond. How many of the blocks are not dirt, stone, wood, or diamond?

Solution:

```
def solution():
  #100000 blocks
  blocks_total = 100000
  #20000 of the blocks are dirt
  blocks_dirt = 20000
  #30000 of the blocks are stone
  blocks_stone = 30000
  #20000 of the blocks are wood
  blocks_wood = 20000
  #30000 of the blocks are diamond
  blocks_diamond = 30000
  #The answer is
  result = blocks_total - blocks_dirt - blocks_stone
    - blocks_wood
  return result
```

Negative Example Reasoning The model correctly defines the necessary variables, but does not subtract the number of diamond blocks from the

total number of blocks.

E.6.3 Appropriateness

Positive Example LeBron James has 12000 points. He is 4000 points away from the all-time scoring record. How many more points does he need to average per game for the next 20 games to break the record?

Negative Example A field hockey team has 11 players. 3 of them are forwards, 3 of them are midfielders, 3 of them are defenders, and 2 of them are goalies. How many forwards are there?

Negative Example Reasoning This question is inappropriate to give to a student because it does not require any mathematical operations to solve. It directly defines the number of forwards on the team.

F Topics for Data Generation

We collected topics and keywords in collaboration with our volunteer annotators with K-12 teaching experience. The topics and keywords span a wide array of student interests including sports, music, video games, movies/TV, animals, vehicles, and food. On top of being manually collected from experts, we believe the resultant list is largely inarguable.

We used a list of 43 topics when comparing models for the experiments reported in Section 4 to make sure the generations could be compared fairly and followed a similar contextual distribution. The full list of topics written in Python list format is below.

Topics : ['Superman', 'Batman', 'Wonder Woman', 'Barbie', 'Power Rangers', 'basketball', 'soccer', 'football', 'volleyball', 'field hockey', 'Fortnite', 'Spiderman', 'Iron Man', 'Captain America', 'Captain Marvel', 'Thor, the God of Thunder', 'Ninja Turtles', 'Black Panther', 'Taylor Swift', 'swimming', 'Pokémon', 'Super Mario', 'Naruto', 'unicorns', 'Hello Kitty', 'Minecraft', 'lacrosse', 'cheer leading', 'LeBron James', 'Steph Curry', 'Patrick Mahomes', 'Serena Williams', 'dogs', 'cats', 'dinosaurs', 'Harry Potter', 'cars', 'planes', 'trains', 'pizza', 'cookies', 'ice cream', 'candy']

G GPT-4 Coding Performance

GPT-4 tends to output additional information beyond what is requested in the prompt and shown in few-shot examples, such as example usage or comments about the Python function solution (see Appendix E.1.2 for an example). This results in the model having a lower percentage of executable code than GPT-3.5 when its output is parsed by the same script we used to parse the output from all other models we evaluated. We used the same parsing script across models to evaluate them all the same way, rather than creating a customized script for GPT-4.