Automatic Generation of Socratic Subquestions for Teaching Math Word Problems

Kumar Shridhar [*♠] **Jakub Macina** [*♠Φ] **Mennatallah El-Assady** [♠Φ]
Tanmay Sinha [▼] **Manu Kapur** [▼] **Mrinmaya Sachan** [♠]

[♠]Department of Computer Science, ETH Zurich
[Φ]ETH AI Center
[▼]Professorship for Learning Sciences and Higher Education, ETH Zurich

Abstract

Socratic questioning is an educational method that allows students to discover answers to complex problems by asking them a series of thoughtful questions. Generation of didactically sound questions is challenging, requiring understanding of the reasoning process involved in the problem. We hypothesize that such questioning strategy can not only enhance the human performance, but also assist the math word problem (MWP) solvers. In this work, we explore the ability of large language models (LMs) in generating sequential questions for guiding math word problem-solving. We propose various guided question generation schemes based on input conditioning and reinforcement learning. On both automatic and human quality evaluations, we find that LMs constrained with desirable question properties generate superior questions and improve the overall performance of a math word problem solver. We conduct a preliminary user study to examine the potential value of such question generation models in the education domain. Results suggest that the difficulty level of problems plays an important role in determining whether questioning improves or hinders human performance. We discuss the future of using such questioning strategies in education.

https://github.com/eth-nlped/scaffolding-generation

1 Introduction

Questioning can be a valuable way of supporting student thinking. It can be conceived as a scaffold (Wood et al., 1976; Quintana et al., 2004), where a more knowledgeable tutor helps a student in solving problems otherwise too difficult. One approach well-suited for mathematics is *funneling* (Wood, 1994), which uses prompting questions to guide students towards a solution.

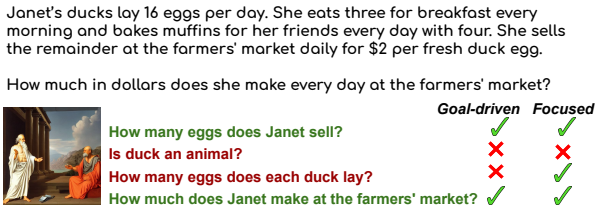

Figure 1: Math word problems can be precedurally solved in multiple reasoning steps. One operationalization of Socratic questioning is to map each step in the procedure to a question. Asking (machines/humans) the right set of questions in a certain sequence (shown in green) can be an effective way to do so. In order to be effective, the Socratic questioning should be focused and goal-driven.

Figure 1 shows an example of a math word problem where this questioning strategy might be beneficial. We hypothesize that these questions can not only help humans in understanding the problem better and improve their performance but can also assist MWP solvers.

Even though question generation (QG) models have been studied for factual SQuAD-like questions (Rajpurkar et al., 2016; Puri et al., 2020), these models fail to generate sequentially-coherent questions (Reddy et al., 2019; Choi et al., 2018). Furthermore, domain-specific questioning is challenging as the QG model needs to understand the reasoning process required to provide fine-grained responses. Moreover, the role of a teacher using questioning is to interject questions that *focus* on the most critical points in an explanation and take the understanding *forward* (Anghileri, 2006). As seen in bold in the Figure 1, we refer later to these properties of questioning as *focused* and *goal-driven*.

In this work, we explore the use of large language models (Raffel et al., 2020; Radford et al., 2019) to generate guiding sub-questions for math word problems. In particular, we use reinforcement learn-

* Equal contribution;
correspondence at: {shkumar, macinaj}@ethz.ch

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 4136 - 4149
December 7-11, 2022

ing (RL) with rewards from various sources including Math question answering (Math QA) models and various forms of input conditioning for generating these questions. We train and evaluate our models on the recently released GSM8K MathQA dataset (Cobbe et al., 2021) of multi-step reasoning MWPs. We illustrate the benefit of our RL-based generation strategy using both automatic and human evaluation metrics. Our evaluation shows that our guided approach makes the generation model ask more *logically relevant* and *structurally correct* questions, which follow the appropriate sequencing of questioning at the *right granularity* level.

We further show that our generated questions, when provided as additional context, can aid a math question answering model, thereby providing further empirical justification of the value of questioning for math QA model training. Questioning could facilitate reasoning of MWP solvers by making intermediate reasoning steps explicit. Finally, we explore the didactic usefulness of our questioning strategy by conducting a preliminary user study and use it to show that the generated sequence of questions may have the potential to improve students' problem-solving. However, we cautiously note that achieving this would require further progress on many fronts in AI and Education.

In what follows, we begin by discussing related work and introducing our research questions in section 2 and section 3. We propose ways to induce these properties in LMs using planning and reinforcement learning in section 4; section 5 empirically demonstrates the effectiveness of inducing questioning strategy in LMs and the quality of generated questions evaluated using automatic metrics and by humans. Finally, we evaluate the potential of using such questions as an educational tool for helping students solve MWPs in section 6.

2 Related Work

Socratic questioning approaches have evolved within the learning sciences community into the *theory of scaffolding* (Wood et al., 1976; Reiser, 2004), which broadly refers to assisting students in problem-solving beyond their zone of proximal development (Quintana et al., 2004). Computer-based scaffolds (e.g., in the form of hints, prompts, feedback) have moderate effects on student learning outcomes (Kim et al., 2018), and our work can be used to automatically generate such scaffolds in the form of questioning prompts. For mathematics, Wood (1994) analyzed interactions in math classrooms and proposed two distinct interaction patterns - *funneling*, which functions by guiding students using leading/prompting questions to a predetermined solution procedure, and *focusing*, which functions by drawing student attention to the critical aspects of the problem. We draw inspiration from this strand of work. Our overall question generation approach can be conceived to be similar to funneling, with specific sub-questions focusing on the important domain concepts.

Research on question generation includes visual question generation (Fan et al., 2018; Wang et al., 2022), generation of questions for student assessment (Stasaski and Hearst, 2017; Wang et al., 2018), generation of factual questions based on Wikipedia articles (Rajpurkar et al., 2016; Ko et al., 2020) or generation of sequential information-seeking questions in dialogue-based scenarios (Reddy et al., 2019; Choi et al., 2018). Other work has also explored similar ideas of improving answerability by question-asking (Klein and Nabi, 2019; Shwartz et al., 2020; Perez et al., 2020; Pan et al., 2021) and ranking them (Rao and Daumé III, 2018). However, factual questions do not usually require much reasoning and mostly boil down to information retrieval from text. In this work, we focus on question generation for reasoning problems.

Prior work on guided and controlled question generation uses either entities as guiding mechanism (Huang et al., 2021) or reinforcement learning-based graph to sequence approach (Chen et al., 2019). Identification of entities and relationships present in the text often uses rule-based or on-shelf extraction tools, which are hard to extend (Dhingra et al., 2020). Often these single-hop questions are combined to form a multi-hop question that requires complex reasoning to solve it (Pan et al., 2021). Controllable text generation has been studied in the past for text generation (Hu et al., 2017; Miladinović et al., 2022; Carlsson et al., 2022), Wikipedia texts (Liu et al., 2018; Prabhumoye et al., 2018) and data-to-text generation (Puduppully and Lapata, 2021; Su et al., 2021). Controlled text generation is particularly useful for ensuring that the information is correct or the numbers are encapsulated properly (Gong et al., 2020). Our task has similar requirements.

A final strand of related work lies in the ballpark of

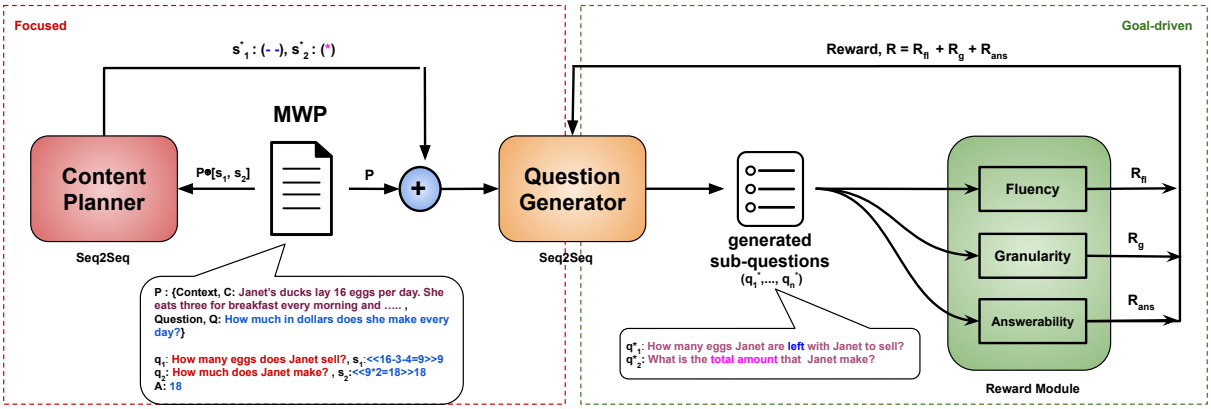

Figure 2: **Our overall methodology:** Two Socratic properties of focused (red dotted box) and goal-driven (green dotted box) question generation are added to the question generation model with a combination of content planning and reward based finetuning. Here, $\oplus$ represents the concatenation operation.

math problem solvers (Hosseini et al., 2014; Kushman et al., 2014; Roy et al., 2015; Seo et al., 2015; Sachan and Xing, 2017; Sachan et al., 2017, 2018, *inter alia*). Recent work in this area uses specialized architectures such as graph-based encoders (Zhang et al., 2020) and tree-based decoders (Xie and Sun, 2019), and more recently, large pretrained LMs which show state-of-the-art results (Cobbe et al., 2021; Shen et al., 2021; Kojima et al., 2022; Wei et al., 2022; Chowdhery et al., 2022). Application of these approaches to the MWP datasets like GSM8K (our data context) still holds considerable room for improvement, primarily in the reasoning capabilities, and the majority of the latest approaches are still unable to solve a lot of the problems and sensitive to even slightest modifications in the problem (Patel et al., 2021; Stolfo et al., 2022; Srivastava et al., 2022) .

3 Research Questions

We now discuss the usefulness of questions in solving a math word problem and then study the different properties of a good questioning strategy.

RQ1: Does sub-questioning help in understanding a math word problem better? Question prompts as a teaching strategy act as instructions that guide the students throughout a problem-solving process (Wood, 1994). Such questioning, as a valid scaffolding strategy (Kim et al., 2018), is valuable in supporting student thinking and is commonplace in high-quality math instruction (Boston and Candela, 2018). We explored the sub-questioning strategy with our trained NLP model and found that sub-questioning helps answer the MWPs more effectively (Table 1). Experiments with NLP models and humans establish the usefulness of sub-questioning in solving MWPs.

RQ2: What are the properties of a good questioning strategy? Once we established that sub-questioning is helpful, we performed the same sub-questioning experiment as RQ1 with NLP models but with the permuted ordering of sub-questions, change in granularity of sub-questions or changed content (Table 2). We observed a decrease in the answering capabilities of the QA model for all the cases, establishing that the right sequence of disciplined questions with relevant content is an essential component of a good questioning strategy. Based on our results and inspired by prior work (Wood, 1994; Anghileri, 2006), we hypothesize the most important components of a Socratic questioning strategy as:

(A) **Focused**: An essential property of a good questioning strategy is to ask questions that are directed towards the most critical domain-specific content. Irrelevant questions not only make the process difficult but also force a diversion in the focus and may increase the cognitive load that a student experiences.

(B) **Goal-driven**: Asking the right sequence of relevant questions that can assist students in reaching the final goal (solving the main question in case of math word problems) is a further important part of good questioning.

4 Methodology

We discuss our approach to modeling Socratic questioning using large LMs. We begin by defining our MWP dataset $\mathcal{D}$ as a collection of MWPs. Each MWP P in the dataset is accompanied by its solution $\mathbf{S}$ and the numerical answer A. We do not

always assume the existence of problem solutions $\mathbf{S}$ and answers A as they can be automatically derived from various MathQA models. Each MWP $P = (C, Q)$ consists of the story context C and the question Q. The problem solution $\mathbf{S}$ consists of n solution steps $\mathbf{S} = (s_1, ..., s_n)$. We define Socratic questioning such that each solution step s_i can be mapped to a sub-question q_i . We refer to $\mathbf{q}$ as a collection of all Socratic questions $q_1, ..., q_n$ for a given MWP P in our work. An example MWP is present in Figure 2.

Our main module is the Question Generator (QG) module, which is a transformer (Vaswani et al., 2017) based encoder-decoder model. The QG model takes the reference Math word problem P and generates the Socratic questions $\mathbf{q}^*$ as close to the true sub-questions $\mathbf{q}$ as possible. The learning objective of the QG module is as:

$$\mathcal{L}_{QG} = -\sum_{i=1}^{n} \log P_{Dec}\left(q_i | q_{:i-1};\ Enc(P)\right) \quad (1)$$

where Enc represents the encoder and Dec represents the decoder for the seq2seq QG model. Note that the sub-questions q_i are decoded word by word in an auto-regressive setting.
Next, we propose to inject the two Socratic questioning properties in our QG model as follows:

4.1 Focused questions

To learn a sequence of disciplined questions focused on specific reasoning steps in the MWP, it is important to ask the right set of questions. We propose a *content planner* ψ that serves as a guiding principle for the QG model to ask the right focused questions. In principle, the content planner module can extract any relevant information to assist the QG model, but for the task of math word problems, we restrict it to operators and equations.[1] Our planning strategies are defined as:

Operators: Given an MWP P, the content planner learns to identify the operations and operators (e.g., addition, multiplication, ..) involved in the problem. Since the operators play a significant role in a given MWP, the generated operators are used as the guiding principle to generate sub-questions by the QG model.

[1]We also do not consider the step-by-step solutions S in our work, as creating step-by-step textual solution requires a lot of time and effort from teachers and even the largest language models fail to understand MWPs easily (Wei et al., 2022; Chowdhery et al., 2022).

Equations: Equations contain important information for an MWP as they involve not just the operators but also the quantities involved in the problem. Similar to operators, equations can play an important guiding principle for asking more focused questions leading towards a correct solution.

We use the same seq2seq architecture for the content planner module as our QG model, with the only difference being that the output comprises a set of equations $s_1^*, .., s_n^*$ or just the operators within the equations (instead of the sub-questions). The generated operators/equations are appended to the input MWP P in the encoder for the QG module and the modified focused learning objective $\mathcal{L}_{QG_f}$ is:

$$\mathcal{L}_{QG_f} = -\sum_{i=1}^{n} \log P_{Dec}\left(q_i | q_{:i-1};\ Enc([P \oplus \texttt{plan}])\right) \quad (2)$$

Here, `plan` depicts the content planner module's output and $\oplus$ depicts the concatenation operation.

4.2 Goal-driven questions

An essential element of a good questioning strategy is to ask goal-driven questions that are not only factually associated to the main problem but also eventually help in answering the main question. However, there can be any number of goal-driven questions that can be asked for a MWP. Thus, our goal is to optimize the questioning strategy such that it is goal-driven, efficient, and rewarding at each step, making sure that the final goal can be achieved with these individual questions. We induce these properties in our QG model using various *rewards* that force the model to stay relevant to the problem. These rewards are defined as:

Fluency: It is important that the generated sub-questions are easily understandable and fluent in the meaning they represent. Although the QG training objective ensures the syntax and semantics of the questions generated, rewarding the system to stay fluent is necessary to remove repetitions and illogical questions.

Granularity: As solving a MWP usually involves multiple reasoning steps, asking relevant questions at each step can help in solving the MWP. Moreover, our questioning strategy is based on the fact that the questions are organised, structured and follow a sequence. With the granularity reward, the model can learn to ask the right number of questions (compared to the number of reasoning steps

to solve MWP) in a specific sequence and refrain from unstructured questions.

Answerability: For every generated question, it is important to evaluate if the generated questions can be answered given context C and can help in answering the overall MWP. We trained an external QA model that can answer the MWPs taking help from the sub-questions and evaluated if the generated question can assist in answering the main problem. The answerability reward is provided on both a step-by-step basis (if the QA model can answer a sub-part of the main problem) and overall (if using all sub-questions, whether the final answer was correct or not).

During training, the QG model samples a set of sub-questions $\mathbf{q}'$, calculates various rewards based on $\mathbf{q}'$. The parameters of the QG model are updated using the REINFORCE algorithm (Williams, 1992) as:

$$\begin{aligned}\mathcal{L}_{\text{RL}} &= -\mathbb{E}_{\mathbf{q}' \sim P_{Dec}}[R(\mathbf{q}, \mathbf{q}', P)] \\ &= -R(\mathbf{q}, \mathbf{q}', P) \sum_{i=1}^{n} \log P_{Dec}\left(q_i | q_{:i-1};\ Enc(P)\right)\end{aligned}$$

The reward function $[R(\mathbf{q}, \mathbf{q}', \mathbf{P})]$ measures the individual rewards for fluency, granularity and answerability and is calculated as:

Fluency: $R_{fl} = BLEU(\mathbf{q}, \mathbf{q}')$

where, BLEU(.,.) represents the BLEU score (Papineni et al., 2002).

Granularity: $R_g = F(\mathbf{q}, \mathbf{q}')$

where, $F(\mathbf{q}, \mathbf{q}') = \mathbf{1} - \frac{||\mathbf{q}| - |\mathbf{q}'||}{|\mathbf{q}'|}$, and $|\mathbf{q}|$ and $|\mathbf{q}'|$ denote the number of questions in $\mathbf{q}$ and $\mathbf{q}'$ respectively.

Answerability: $R_{ans} = F(A, A')$

where, $F(A, A') = 1$ if the final answer from the QA model is correct when it is given sub-questions $\mathbf{q}'$ alongside the MWP $\mathbf{P}$, and 0 otherwise. A' denotes the answer from the QA model and A denotes the true answer.

We also evaluated the step-by-step performance of the QA model on the generated sub-questions to check if the QA model can answer the generated sub-questions correctly. This allows us to provide partial rewards at each step of the generation model. The modified sub-step answerability reward is $F(A, A') = \frac{\#a'}{|\mathbf{q}'|}$, where $\#a'$ and $|\mathbf{q}'|$ denote the number of correct answers to the generated sub-questions and total number of generated questions respectively.

Variation	GPT-2	GPT-3
P	5.45 (↓ 47%)	29 (↓ 38%)
P $\oplus$ $\{q\}$	**10.46**	**47**

Table 1: Comparison of Math QA accuracy (in %) with and without Socratic questions for GSM8K test dataset. (↓) represents the drop in the accuracy when compared to the Socratic questions (P $\oplus$ $\{q\}$). $\oplus$ represents the concatenation operation. GPT-2 model was trained with and without Socratic questions while GPT-3 model (Brown et al., 2020) was prompted using one-shot example (more details in Appendix subsection B.2).

4.3 Overall Loss Function

Finally, with the induced Socratic properties in the QG model, the total loss is defined as a combination of the focused learning loss $\mathcal{L}_{QG_f}$ and the loss of the rewards $\mathcal{L}_{RL}$, as:

$$\mathcal{L} = \alpha\, \mathcal{L}_{QG_f} + (1 - \alpha)\, \mathcal{L}_{RL} \tag{3}$$

where α is a weighting factor.

5 Empirical Analysis

We now demonstrate the effectiveness of inducing the defined questioning properties in large LMs.

Dataset We study the properties of Socratic questioning on the GSM8K dataset[2] (Cobbe et al., 2021) that consists of 8.5K grade school math word problems. Each problem requires 2 to 8 reasoning steps to solve, and solutions primarily involve a sequence of elementary calculations using basic arithmetic operations ($+ - * /$). The dataset is segmented into 7.5K training problems and 1K test problems.

Models We used T5 (Raffel et al., 2020) as the backbone of both our QG and content planning modules. For reward generating QA model, we used GPT-2 (Radford et al., 2019) for all RQ2 experiments because of resource constraints. However, a better QA model like GPT-3 (Brown et al., 2020) can be used in the future. Both QG and content planning models are fine-tuned on the GSM8K training set using the Huggingface library (Wolf et al., 2020).

[2]https://github.com/openai/grade-school-math

Variation	QA Accuracy
Granularity	
P $\oplus$ $\{q\}^0$	5.45 ($\downarrow$ 45%)
P $\oplus$ $\{q\}^{0.25}$	3.94 ($\downarrow$ 62%)
P $\oplus$ $\{q\}^{0.5}$	3.35 ($\downarrow$ 67%)
P $\oplus$ $\{q\}^{0.75}$	9.70 ($\downarrow$ 7%)
P $\oplus$ $\{q\}^1$	**10.46**
Order	
P $\oplus$ shuffle($\{q\}$)	8.94 ($\downarrow$ 14%)
Relevance	
P $\oplus$ <base-ques>	2.57 ($\downarrow$ 75%)

Table 2: Comparison of Math QA accuracy (in %) for different variations of experiments with ground truth data. $\{q\}^k$ represents that only k% of the ground truth sub-questions are used and selected randomly. For e.g., $\{q\}^{0.25}$ represents only 25% of the sub-questions are used. shuffle($\{q\}$) represents all sub-questions, but with shuffled order. Finally, `<base-ques>` are the sub-questions generated from a T5 large model without fine-tuning on our task. ($\downarrow$) represents the drop in the accuracy when compared to the Socratic questions (P $\oplus$ $\{q\}$). $\oplus$ represents the concatenation operation. GPT-2 small was used as QA model for all the above experiments.

Implementation Details For the training of the models, we used Nvidia Tesla A100 with 40 GB of GPU memory. We ran each experiment for 50 epochs, with a periodical evaluation of the validation set. Training time without using rewards is 10 minutes per epoch. With rewards, the training time per epoch is increased to several hours. We used the T5-large model without modifications for the content planner and question generation module and GPT-2 small as the QA solver.

Evaluation Metrics We report automatic evaluation using SacreBLEU (Post, 2018) which is based on exact word overlap, BERT F1 score (Zhang et al., 2019) which is based on DeBERTa (He et al., 2020) as the similarity model. We also report $\#Q$, the number of questions generated compared to the number of ground truth reasoning steps (same as *Granularity* reward), and Math QA Solver accuracy (same as the overall *Answerability* reward) to assess if our generated questions helped the QA model reach the final numerical solution.

5.1 RQ1: Does sub-questioning help in understanding math concepts better?

We hypothesize that high-quality sub-questioning helps Math QA solvers to reach the correct solution, especially when questions are relevant to the concept to be learnt, in the right sequence (ordering) with high granularity in their structure. We verify our hypothesis with GPT-2 model as a QA solver after fine-tuning it on the training set of the GSM8K dataset and the GPT-3 model with one-shot prompting. Table 1 demonstrates that the Socratic questioning improves the performance of the QA solver as high as 45%. Then, we vary the properties of the test questions and examine the performance of the QA Solver. Table 2 demonstrates that Socratic questions significantly improve the model performance from 5.45% to 10.46%. Sub-questioning even helps when only 75% Socratic questions are retained (denoted as $\{q\}^{0.75}$ in the table) or when the order is shuffled (this might be an artefact of the dataset containing a minority of examples with strict order). An interesting observation is that when the number of Socratic questions is reduced by half or lower (while preserving their order), the model gets confused and performs worse than when it had no sub-questions. Finally, we take the pre-trained T5 model and without fine-tuning it for our task, we take the outputs and used it alongside the problem P as additional information to solve the problem. The performance goes as low as 2.57%, indicating that non-relevant information degrades the performance.

Planning	BLEU	BERT F1	#Q
None	51.53	0.783	0.428
Operators	54.98	0.788	0.642
+ planner	45.05	0.779	0.346
Equations	**58.82**	**0.813**	**0.807**
+ planner	52.48	0.787	0.485

Table 3: **Focused questions**: QG model performance compared on the gold set of ground truth test questions with different planning strategies. Note that for the planner rows, content planning information from the ground truth data is replaced with the output from the content planner model.

5.2 RQ2: What are the properties of a good questioning strategy?

We now present our analysis on inducing the two Socratic properties to LMs.

Focused generation: Table 3 compares the two planning strategies. Results demonstrate that planning strategies improve the baseline methods by more than 3% on BLEU score with operators as planning, and by more than 7% with equations.

Strategy	BLEU	BERT F1	#Q
Baseline	13.02	0.566	0.056
Fine-tuned	51.53	0.783	0.428
+ fluency	52.21	0.784	0.440
+ # of questions	51.86	0.784	0.431
+ QA	52.22	0.783	0.417
+ all weighted	53.39	0.781	0.431
Eq planning	58.82	0.813	0.807
+ fluency	59.52	**0.816**	**0.818**
+ # of questions	**59.75**	0.814	0.811
+ QA	59.37	0.813	0.799
+ all weighted	59.62	0.815	0.815

Table 4: **Goal-driven questions**: QG model performance compared to the gold set of ground truth questions with different rewards.

Similar to the BLEU score, we achieve better performance on BERT F1 scores too. Finally, the number of correct question count improves with planning and doubles compared to the no-planning variant. However, results show that in all the variants the number of generated sub-questions is less than the number of reasoning steps. This could be improved further by oversampling during the beam search (beam search settings are the same for all variants in this experiment). The results degrade when the ground truth content (both equations and planning) is replaced by our content planner module. This is expected as the errors in the content planning module are cascaded when generating sub-questions. However, with more powerful models, errors in the content planner can be reduced, leading to improvement in all the metrics. See the Appendix for experiments with the iterative splitting of MWP into multiple parts for generation.

Goal-driven generation: Table 4 summarizes the results for the rewards as a strategy to incentivize the model to generate goal-driven and rewarding questions. We can observe the gains associated with each reward for both the baseline model and the best-performing model from Table 3 (equation-based content planning model in our case), suggesting the importance of rewards.

QA performance We study the impact of the QG model considering both Socratic properties as shown in Table 5. Sub-questions with operators and equations as planning improves the QA performance by $1-2\%$. Rewards, although improves the QG quality, have a negligible effect on QA performance. This is mainly because slight improvement in sub-questions quality does not necessarily help in reaching the final goal.

Strategy	QA Accuracy
No planning	6.74
+ rewards	6.75
Operators	7.50
+ rewards	7.52
Equations	8.49
+ rewards	8.50

Table 5: Overall model variation and the influence on Math QA solver accuracy (in %) with different planning and reward strategies. Here, GPT-2 small is used as the QA model. Please note upper limit using ground truth questions is 10.46% as shown in Table 1.

Planning	BLEU	BERT F1	#Q
None	51.53	0.783	0.428
Diff op, diff #	51.59	0.785	0.415
Diff op, same #	54.26	0.786	0.546
Operators (op)	**54.98**	**0.788**	**0.642**

Table 6: Manipulating the planning inputs influences the quality of generated questions and overall QG model performance. **same #** has the same number of operators as number of reasoning steps but the types (+-/*) are shuffled, **diff #** has both number and type of operator shuffled.

5.3 Human quality evaluation

Next, we perform a human evaluation of the questions generated for 100 randomly selected test MWPs to assess the quality of our model generation (our best model) compared to the baseline (with no planning or reward-based strategies). For this analysis, we divided the questions among 4 annotators with an overlap of 40% of the questions among them[3] to evaluate the generated question quality on the following factors. A 5-point Likert scale ranging from 1 (poor) to 5 (very good) was used for each dimensions of quality assessment: *repetition* - whether questions are repeated, *factuality* - whether all questions can be solved by the information given in the problem, *logical relevance* - if the question is logically related to the MWP, *right sequence* - correct sequence of questions leading to the final answer, *granularity* - questions are granular enough to solve the problem but are still relevant and no retrieval or basic common sense

[3]Overlap allows us to compute inter-annotator agreement.

questions are asked, *completeness* - questions are complete with all steps covered to reach to the final answer, and *fluency* - grammatical correctness and fluent in the language.

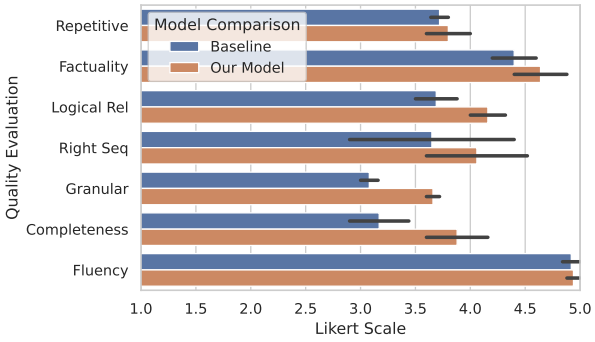

Figure 3: Comparison of baseline versus our model generated sub-questions on several metrics from our human evaluations (showing mean and standard deviation).

Figure 3 presents our findings, clearly demonstrating that our planning and reward strategies lead to superior quality questions on the MWP task. Although both baselines and our model-generated text achieve almost full score (5) on the fluency parameter, our model-generated questions are more aligned to the MWP, thus leading to a higher score on all the other parameters. We also present a randomly selected sample of generated questions in the Appendix.

5.4 Ablation study: Manipulating question properties

Both planning strategies help generate better questions. To gain a deeper understanding of how content planner ψ affects generated questions, we further analyze the influence of operators as a planning strategy. Here, we randomize operators and their sequence and measure change in performance. Table 6 shows that the correct sequence of operators with the correct number of operators guides the generation process better than randomized versions. A gap between the correct count of operators and random count indicates that having a correct number of operators (of any type) is more valuable than the exact type of operators. We observed that the number of operators guides the model in terms of the number of questions that need to be asked, while type changes the overall quality. Needless to say, for the same number of operators, quality matters.

6 A preliminary user study with learners

Finally, we designed a preliminary user study to evaluate whether our generated questions, when presented as further problem-solving exercises (as typically used in educational settings) can help learners on the way to solving the overall problem. Given our research question, we hypothesized that *guidance with questions can increase the overall problem-solving success rate for users in the questions (treatment) group compared to the no-questions control group.* Our study uses Socratic questions as the main pedagogical intervention. We focus on participants who cannot solve a problem on the first attempt to clearly distinguish the impact of automated sub-questioning. The key metric we measure is the success rate, which is defined as the percentage of correctly solved problems.

For our study, we built a simple user interface which allowed participants to solve math word problems (see Figure 5 and Figure 6 in the appendix). The interface contained a calculator which the users could use if needed. The study comprises 5 pre-test problems and 8 problem-solving exercises. These problems were randomly selected from the GSM8K test set. Our user study with this interface was then deployed on Mechanical Turk and participants were hired using the platform and were paid 10-12$ per hour. We selected participants with moderate levels of prior knowledge using the pre-test scores as the selection criteria, and only those scoring in the range of 40-80% were selected for the study. This way, we excluded both low-prior knowledge participants and experts in our study to ensure there was a learning possibility.

We randomly split the participants into two groups - *no-questions group* ($N = 19$) with no question prompts, and *questions group* ($N = 17$) with questions generated from our model. Both groups used the same interface for solving math word problems and had the opportunity to resolve their answers after the first incorrect submission. The only difference was that after incorrectly solving a problem on the first submission, participants in the questions group saw sub-questions, while those in the no-questions group were only prompted to try again. The sub-questions were generated using the best-performing model with planning and rewards.

The results of the user study are shown in Table 7. The first attempt success rate is 58.4% for the

control group and 66.0% for the treatment group, which might be the result of a slightly skewed prior knowledge distribution of 0.68 and 0.65 for treatment and control groups respectively. Even though participants in the treatment group ($M = 124.9$, $SD = 92.1$) spend significantly more time ($p < 0.01$) solving problems during the second attempt relative to the control group ($M = 41.5$, $SD = 31.4$), we did not find any statistically significant difference between the groups in the second submission success rate ($p = 0.659$, $BF_{01} = 2.755$, Cohen's $d = 0.157$), indicating weak odds favouring the null hypothesis and rather a small effect size.

As our study was unable to establish overall performance improvements, we further analysed the second submission success rate per problem (see Figure 4), and correlated it with the difficulty of the question. This analysis indicated that sub-questioning seems to improve the success of simpler problems and degrade the accuracy for relatively more complex problems. Prior work has suggested that the effectiveness of question prompts varies according to an individual's prior knowledge (Kim et al., 2018), and with insufficient prior knowledge, performance for complex problems may suffer. A posthoc inspection of the generated sub-questions for more complex problems shows that they also scored lower in the human quality evaluation. Thus, we hypothesize that for more complex questions, the generated sub-questions are not good enough, and so they may make the task more challenging for participants.

While we were not able to establish any direct benefits of automatic Socratic questioning in a real learning scenario, we leave a more complete user study for future work. Deployment of Socratic questioning systems in real educational scenarios would require a better assessment of question generation quality as well as a better understanding of learners. We believe this is an interesting avenue for future research and encourage future work to attempt to address these issues.

7 Conclusion

We study the importance of sub-questioning for learning a mathematical concept and explore how LMs may generate these sub-questions. We demonstrate the usefulness of Socratic questioning strategies and propose ways to induce these properties

Group	1st success		2nd success	
	M	SD	M	SD
No-questions	58.4	23.0	35.8	32.5
Questions	66.0	21.1	31.0	27.9

Table 7: User study success rates (in %) before after introduction of sub-questions. **1st success** is the proportion of exercises solved correctly on the first attempt and **2nd success** is the proportion of correctly solved exercises on the second attempt (out of all incorrectly solved on the first attempt).

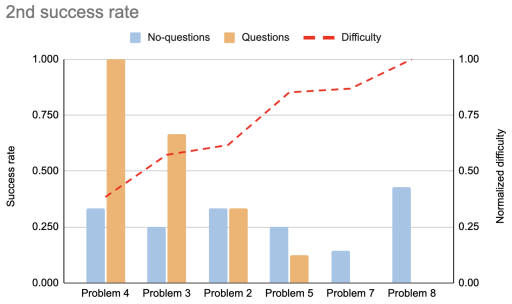

Figure 4: Second submission success rate for problems with at least 10% occurrence for each group (excluding the two simplest problems 1 and 6). Difficulty level is annotated blind to the correct solution.

in LMs. We further evaluate if these questions can assist students in learning domain concepts. We found that the generated questions were generic for each student and if adapted to their prior knowledge and intermediate solutions, their effectiveness could have been greater.

A discussion on limitations of our work

Our questioning strategy, although utilizes information from the content planner and the reward strategy, leaves much to be desired in terms of its controllability. Based on our user study, we need to be careful in using the questioning strategy in real educational contexts, as improper content can sometimes do more harm than good. Based on the prior work, we focused on two aspects of goodness in questioning in math education. However, this is not a complete list and other aspects could also be important. We note that our user study was focused on the intermediate success rate rather than on actual learning. From a learning standpoint, asking questions that are always easily answerable won't lead to deeper, wider learning. If learners do not have to struggle to answer the sub-questions being asked and are instead repeating something verbatim or offering a slightly reconfigured version of

what they have been asked, they are probably answering sub-questions that do not require conceptual understanding. Another limitation of our work is that our user study was underpowered due to resource constraints, which prevents us from drawing strong conclusions at this point. A larger user study is however forthcoming.

Finally, we choose to focus on Socratic questioning in a rather narrow sense of trying to call learners' attention to relevant facts and then implicitly stimulating them to integrate facts and draw conclusions. However, when taken together with all its nuances, the effectiveness of Socratic questioning can be posited to depend on other critical question types that seek clarification (e.g., can you rephrase?), evidence (e.g., can you provide an example?) and implication (e.g., why do you think. . . ?) from learners too, all of which are truly dialogic and naturally leave room for learner questions. When both the teacher and learners are jointly responsible for pushing the dialogue forward, intermediate success may also not always be desirable as learner errors and misconceptions may offer an important hook for the teacher to nudge the dialogue productively.

Acknowledgements

This project was made possible by an ETH AI Center Doctoral Fellowship to Jakub Macina with partial support by the Asuera Stiftung and the ETH Zurich Foundation. Many thanks to the group members and our reviewers for their valuable feedback.

References

Julia Anghileri. 2006. Scaffolding practices that enhance mathematics learning. *Journal of Mathematics Teacher Education*, 9(1):33–52.

Melissa D Boston and Amber G Candela. 2018. The instructional quality assessment as a tool for reflecting on instructional practice. *ZDM*, 50(3):427–444.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Fredrik Carlsson, Joey Öhman, Fangyu Liu, Severine Verlinden, Joakim Nivre, and Magnus Sahlgren. 2022. Fine-grained controllable text generation using non-residual prompting. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6837–6857, Dublin, Ireland. Association for Computational Linguistics.

Yu Chen, Lingfei Wu, and Mohammed J. Zaki. 2019. Reinforcement learning based graph-to-sequence model for natural question generation. *CoRR*, abs/1908.04942.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wentau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184, Brussels, Belgium. Association for Computational Linguistics.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Bhuwan Dhingra, Manzil Zaheer, Vidhisha Balachandran, Graham Neubig, Ruslan Salakhutdinov, and William W. Cohen. 2020. Differentiable reasoning over a virtual knowledge base. *CoRR*, abs/2002.10640.

Zhihao Fan, Zhongyu Wei, Piji Li, Yanyan Lan, and Xuanjing Huang. 2018. A question type driven framework to diversify visual question generation. In *IJCAI*, pages 4048–4054.

Heng Gong, Wei Bi, Xiaocheng Feng, Bing Qin, Xiaojiang Liu, and Ting Liu. 2020. Enhancing content planning for table-to-text generation with data understanding and verification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2905–2914, Online. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Mohammad Javad Hosseini, Hannaneh Hajishirzi, Oren Etzioni, and Nate Kushman. 2014. Learning to solve arithmetic word problems with verb categorization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 523–533.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P. Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1587–1596. JMLR.org.

Qingbao Huang, Mingyi Fu, Linzhang Mo, Yi Cai, Jingyun Xu, Pijian Li, Qing Li, and Ho-fung Leung. 2021. Entity guided question generation with contextual structure and sequence information capturing. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35:13064–13072.

Nam Ju Kim, Brian R Belland, and Andrew E Walker. 2018. Effectiveness of computer-based scaffolding in

the context of problem-based learning for stem education: Bayesian meta-analysis. *Educational Psychology Review*, 30(2):397–429.

Tassilo Klein and Moin Nabi. 2019. Learning to answer by learning to ask: Getting the best of gpt-2 and bert worlds. *arXiv preprint arXiv:1911.02365*.

Wei-Jen Ko, Te-yuan Chen, Yiyan Huang, Greg Durrett, and Junyi Jessy Li. 2020. Inquisitive question generation for high level text comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6544–6555, Online. Association for Computational Linguistics.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*.

Nate Kushman, Yoav Artzi, Luke Zettlemoyer, and Regina Barzilay. 2014. Learning to automatically solve algebra word problems. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 271–281.

Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. *arXiv preprint arXiv:1801.10198*.

Đorđe Miladinović, Kumar Shridhar, Kushal Jain, Max B Paulus, Joachim M Buhmann, and Carl Allen. 2022. Learning to drop out: An adversarial approach to training sequence vaes. *arXiv preprint arXiv:2209.12590*.

Liangming Pan, Wenhu Chen, Wenhan Xiong, Min-Yen Kan, and William Yang Wang. 2021. Unsupervised multi-hop question answering by question generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5866–5880, Online. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. 2021. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, Online. Association for Computational Linguistics.

Ethan Perez, Patrick Lewis, Wen-tau Yih, Kyunghyun Cho, and Douwe Kiela. 2020. Unsupervised question decomposition for question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8864–8880, Online. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W Black. 2018. Style transfer through back-translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 866–876, Melbourne, Australia. Association for Computational Linguistics.

Ratish Puduppully and Mirella Lapata. 2021. Data-to-text generation with macro planning. *Transactions of the Association for Computational Linguistics*, 9:510–527.

Raul Puri, Ryan Spring, Mohammad Shoeybi, Mostofa Patwary, and Bryan Catanzaro. 2020. Training question answering models from synthetic data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5811–5826, Online. Association for Computational Linguistics.

Chris Quintana, Brian J. Reiser, Elizabeth A. Davis, Joseph Krajcik, Eric Fretz, Ravit Golan Duncan, Eleni Kyza, Daniel Edelson, and Elliot Soloway. 2004. A scaffolding design framework for software to support science inquiry. *Journal of the Learning Sciences*, 13(3):337–386.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Sudha Rao and Hal Daumé III. 2018. Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2737–2746, Melbourne, Australia. Association for Computational Linguistics.

Siva Reddy, Danqi Chen, and Christopher D Manning. 2019. Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

Brian J. Reiser. 2004. Scaffolding Complex Learning: The Mechanisms of Structuring and Problematizing Student Work. *Journal of the Learning Sci-*

ences, 13(3):273–304. Publisher: Routledge _eprint: https://doi.org/10.1207/s15327809jls1303_2.

Subhro Roy, Tim Vieira, and Dan Roth. 2015. Reasoning about quantities in natural language. *Transactions of the Association for Computational Linguistics*, 3:1–13.

Mrinmaya Sachan, Kumar Dubey, and Eric Xing. 2017. From textbooks to knowledge: A case study in harvesting axiomatic knowledge from textbooks to solve geometry problems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 773–784.

Mrinmaya Sachan, Kumar Avinava Dubey, Tom M Mitchell, Dan Roth, and Eric P Xing. 2018. Learning pipelines with limited data and domain knowledge: A study in parsing physics problems. *Advances in Neural Information Processing Systems*, 31.

Mrinmaya Sachan and Eric Xing. 2017. Learning to solve geometry problems from natural language demonstrations in textbooks. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (* SEM 2017)*, pages 251–261.

Minjoon Seo, Hannaneh Hajishirzi, Ali Farhadi, Oren Etzioni, and Clint Malcolm. 2015. Solving geometry problems: Combining text and diagram interpretation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1466–1476, Lisbon, Portugal. Association for Computational Linguistics.

Jianhao Shen, Yichun Yin, Lin Li, Lifeng Shang, Xin Jiang, Ming Zhang, and Qun Liu. 2021. Generate & rank: A multi-task framework for math word problems. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2269–2279, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Vered Shwartz, Peter West, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Unsupervised commonsense question answering with self-talk. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4615–4629, Online. Association for Computational Linguistics.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.

Katherine Stasaski and Marti A. Hearst. 2017. Multiple choice question generation utilizing an ontology. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 303–312, Copenhagen, Denmark. Association for Computational Linguistics.

Alessandro Stolfo, Zhijing Jin, Kumar Shridhar, Bernhard Schölkopf, and Mrinmaya Sachan. 2022. A causal framework to quantify the robustness of mathematical reasoning with language models. *arXiv preprint arXiv:2210.12023*.

Yixuan Su, David Vandyke, Sihui Wang, Yimai Fang, and Nigel Collier. 2021. Plan-then-generate: Controlled data-to-text generation via planning. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 895–909, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Ruonan Wang, Yuxi Qian, Fangxiang Feng, Xiaojie Wang, and Huixing Jiang. 2022. Co-VQA : Answering by interactive sub question sequence. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2396–2408, Dublin, Ireland. Association for Computational Linguistics.

Zichao Wang, Andrew S Lan, Weili Nie, Andrew E Waters, Phillip J Grimaldi, and Richard G Baraniuk. 2018. Qg-net: a data-driven question generation model for educational content. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*, pages 1–10.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *CoRR*, abs/2201.11903.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8(3–4):229–256.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

David Wood, Jerome S Bruner, and Gail Ross. 1976. The role of tutoring in problem solving. *Child Psychology & Psychiatry & Allied Disciplines*.

Terry Wood. 1994. Patterns of interaction and the culture of mathematics classrooms. In *Cultural perspectives on the mathematics classroom*, pages 149–168. Springer.

Zhipeng Xie and Shichao Sun. 2019. A goal-driven tree-structured neural model for math word problems. In *IJCAI*, pages 5299–5305.

Jipeng Zhang, Lei Wang, Roy Ka-Wei Lee, Yi Bin, Yan Wang, Jie Shao, and Ee-Peng Lim. 2020. Graph-to-tree learning for solving math word problems. Association for Computational Linguistics.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

A Details of User Study

We perform a user study using Amazon Mechanical Turk. Participants which did not spend a minimum time per question were excluded from the analysis. Generated questions used in the questions group are listed in the Table 9.

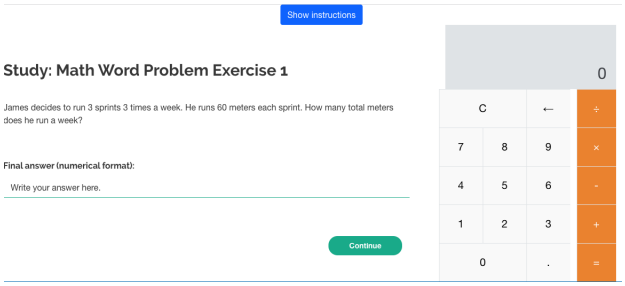

Figure 5: Interface for our user study (cf. Section 6). For each problem, the first screen contains the MWP text, a calculator, and an input box to submit the answer.

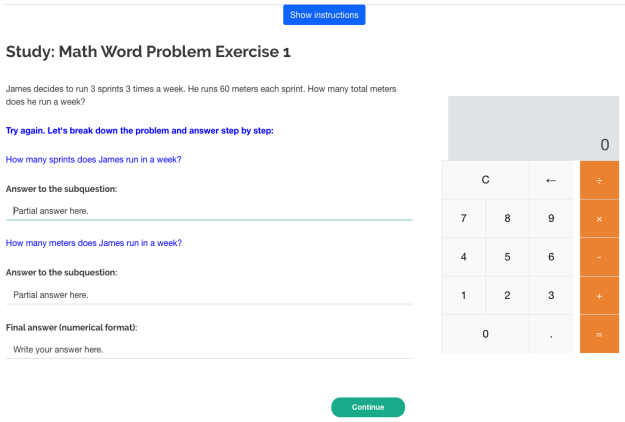

Figure 6: After submitting an incorrect solution on the first attempt in the treatment group, our model generated sub-questions are shown to the participants to guide them through the problem-solving process. The control group only sees a prompt to try again.

Planning	BLEU	BERT F1	#Q
None	49.39	0.763	0.390
Operators	55.25	0.779	0.752
Equations	58.31	0.795	0.819

Table 8: QG model performance compared on the gold set of ground truth test questions with different planning strategies in an iterative setting.

B Experimental details

B.1 Iterative

Except global strategy to generate questions given a MWP, we experimented with iteratively generation on the sentence level.

This also explains the $\#Q$ for iterative case to be not equal to 1 as there are some duplicates generated by the model and sometimes the split is not perfect.

B.2 GPT-3 prompting

We used one shot prompting for GPT-3 meaning we provide one example (Q,A) to the model and let it predict the answer (A) for the next question (Q) provided.

No sub-questions Problem: John has 10 hectares of a pineapple field. There are 100 pineapples per hectare. John can harvest his pineapples every 3 months.
Q: How many pineapples can John harvest within a year? **A**: John has 100 x 10= «100*10=1000»1000 pineapples on his field. John can harvest his Pineapple 12 / 3 = «12/3=4»4 times per year. Therefore John can harvest 1000 x 4 = «1000*4=4000»4000 pineapples per year. #### 4000

Socratic sub-questions Problem: John has 10 hectares of a pineapple field. There are 100 pineapples per hectare. John can harvest his pineapples every 3 months.
Q: How many pineapples does John have? **A**: John has 100 x 10= «100*10=1000»1000 pineapples on his field.
Q: How many times can John harvest his pineapples? **A**: John can harvest his Pineapple 12 / 3 = «12/3=4»4 times per year.
Q: How many pineapples can John harvest within a year? **A**: Therefore John can harvest 1000 x 4 = «1000*4=4000»4000 pineapples per year. #### 4000

C Error analysis of generated questions

Table 9 studies some of the errors encountered by us in our question generation strategy.

Id	Text	Generated questions
1	James decides to run 3 sprints 3 times a week. He runs 60 meters each sprint. How many total meters does he run a week? (540)	How many sprints does James run in a week? How many meters does James run in a week?
2	Carla is downloading a 200 GB file. Normally she can download 2 GB/minute, but 40% of the way through the download, Windows forces a restart to install updates, which takes 20 minutes. Then Carla has to restart the download from the beginning. How long does it take to download the file? (160)	**Missing: time to download 40% of the file** How many minutes does it take to install updates? How many minutes does it take to restart the download? How many minutes does it take to download the file?
3	John buys twice as many red ties as blue ties. The red ties cost 50% more than blue ties. He spent $200 on blue ties that cost $40 each. How much did he spend on ties? (800)	How many blue ties did John buy? How many red ties did John buy? How much did the red ties cost? How much did John spend on ties?
4	Mr. Tan sold his house for $400 000. He paid the transfer fees that amount to 3% of the selling price and also paid a brokerage fee that is 5% of the selling price. If he also paid $250 000 for the remaining loan amount of the house, how much is Mr. Tan's net proceeds from selling the house? (118000)	How much did Mr. Tan pay in transfer fees and brokerage fees? How much did Mr. Tan pay in total? How much is Mr. Tan's net proceeds from selling the house?
5	John drives for 3 hours at a speed of 60 mph and then turns around because he realizes he forgot something very important at home. He tries to get home in 4 hours but spends the first 2 hours in standstill traffic. He spends the next half-hour driving at a speed of 30mph, before being able to drive the remaining time of the 4 hours going at 80 mph. How far is he from home at the end of those 4 hours? (45)	How far did John drive in the first 3 hours? How far did John drive in the remaining 2 hours? How fast did John drive in the next half-hour? How fast did John drive in the remaining 2 hours? **Error: remaining 1 and half hour.** How far did John drive in those 4 hours? How far is John from home at the end of those 4 hours?
6	Charlie wants to sell beeswax candles. For every pound of beeswax, he can make 10 tapered candles. One pound of beeswax and the wicks cost $10.00 in supplies. If he sells each candle for $2.00 each, what is his net profit if he makes and sells 20 candles? (20)	How many pounds of beeswax does Charlie need? How much will each candle cost? How much will Charlie sell the candles for? **Missing: selling price for 20 candles.** What is Charlie's net profit?
7	Shiela bought five cell phones for $150 each for a 3-month installment. A 2% interest will be charged for each unit. How much in total will Shiela pay each month for the period of 3 months? (255)	How much is the interest? How much will Shiela pay in total? **Missing: question about the price of 5 cell phones.** How much will Shiela pay each month for 3 months?
8	Melanie is a door-to-door saleswoman. She sold a third of her vacuum cleaners at the green house, 2 more to the red house, and half of what was left at the orange house. If Melanie has 5 vacuum cleaners left, how many did she start with? (18)	**Wrong order: 1) before visiting orange house?, 2) before visiting red house?, 3)before visiting green house?** How many vacuum cleaners did Melanie sell at the green house? How many vacuum cleaners did Melanie sell at the red house? How many vacuum cleaners did Melanie have left after selling to the red house? How many vacuum cleaners did Melanie start with?

Table 9: User study problems and generated sub-questions