Automatic Generation of Question Hints for Mathematics Problems using Large Language Models in Educational Technology

Junior Cedric Tonga[*,1,2] JUNIORTONGA2022@GMAIL.COM
Benjamin Clement[2] BENJAMIN-C@EVIDENCEB.COM
Pierre-Yves Oudeyer[2,3] PIERRE-YVES.OUDEYER@INRIA.FR
[1]*ENS Paris-Saclay, France*
[2]*EvidenceB, France*
[3]*Inria Bordeaux, France*

Abstract

The automatic generation of hints by Large Language Models (LLMs) within Intelligent Tutoring Systems (ITSs) has shown potential to enhance student learning. However, generating pedagogically sound hints that address student misconceptions and adhere to specific educational objectives remains challenging. This work explores using LLMs (GPT-4o and Llama-3-8B-instruct) as teachers to generate effective hints for students simulated through LLMs (GPT-3.5-turbo, Llama-3-8B-Instruct, or Mistral-7B-instruct-v0.3) tackling math exercises designed for human high-school students, and designed using cognitive science principles. We present here the study of several dimensions: 1) identifying error patterns made by simulated students on secondary-level math exercises; 2) developing various prompts for GPT-4o as a teacher and evaluating their effectiveness in generating hints that enable simulated students to self-correct; and 3) testing the best-performing prompts, based on their ability to produce relevant hints and facilitate error correction, with Llama-3-8B-Instruct as the teacher, allowing for a performance comparison with GPT-4o. The results show that model errors increase with higher temperature settings. Notably, when hints are generated by GPT-4o, the most effective prompts include prompts tailored to specific errors as well as prompts providing general hints based on common mathematical errors. Interestingly, Llama-3-8B-Instruct as a teacher showed better overall performance than GPT-4o. Also the problem-solving and response revision capabilities of the LLMs as students, particularly GPT-3.5-turbo, improved significantly after receiving hints, especially at lower temperature settings. However, models like Mistral-7B-Instruct demonstrated a decline in performance as the temperature increased. This study advances our understanding of the potential and limitations of LLMs in educational contexts, towards integrating these models into pedagogically grounded.

Keywords: Large Language Models, educational technologies, mathematics, hint generation, question generation, pedagogical stance.

1. Introduction

Digital education has gained popularity over the last decade, highlighting the importance of Intelligent Tutoring Systems (ITSs). These systems are seen as essential tools to address specific educational challenges, such as the need for personalized learning in a system often

[*] Work done during internship at EvidenceB.

reliant on pedagogical teaching and standardized tests, the inaccessibility of private tutoring for everyone, and the difficulty in finding expert tutors at reasonable costs (Bray, 1999; Graesser et al., 2012). The key feature of ITS is their ability to provide step-by-step guidance to students while they work on problems, with hints playing a crucial role in their capacity to offer this assistance (Kinnebrew et al., 2015). In the educational context, hints refer to pedagogical questions or suggestions given to learners to help them solve problems, answer questions, or complete tasks. Previous research has shown that providing immediate automated feedback to students within ITS can improve learning outcomes (Kochmar et al., 2020; Razzaq et al., 2020).

However, designing such systems remains a challenge. Indeed, a system that directly gives the correct answer when the learner is wrong, which may occurs with Large Language Models (LLMs), does not encourage any effort and can diminish engagement (Nie et al., 2024). While a system that recognizes the learner's incorrect attempt and provides informative hints related to the learner's existing knowledge encourages critical thinking, problem-solving skills, and independent learning. The challenge to develop such system resides in particular in meeting the diverse learning needs of students and fostering a deeper understanding of complex concepts. These systems can leverage recent advances in Natural Language Processing (NLP), generative AI, and LLMs such as the GPT family models (ChatGPT (noa, a)) or Mistral (Jiang et al., 2023), Llama (Touvron et al., 2023a), to be enhanced by integrating LLMs. However, to achieve such a system based on LLMs, these models must meet a wide range of requirements, such as understanding the question and why the student's answer is incorrect, particularly in mathematics, which is the focus of our study, as well as being aligned with educational goals, pedagogical theory, and cognitive processes. By cognitive processes, we mean the skills we aim to develop, the challenges to include in the exercise, and the potential biases in the student's understanding.

In this study, we investigate the application of LLMs to generate effective hints for simulated students solving math problems. Figure 1 illustrates the overall approach adopted in this paper. These problems are designed for human students at the high-school level, and are grounded in cognitive science principles (Knops, 2022; Gros et al., 2020). Since hint generation involves using LLMs, it is crucial to first understand how these models perform in generating hints when used with simulated students before applying them to real-world scenarios. LLMs can simulate human behaviors, as demonstrated by (Markel et al., 2023), who built AI students and studied their interactions with human tutors. Accordingly, we have chosen to use LLMs to simulate students and teacher in our experiments. The mains contribution of this work are: 1) the evaluation of GPT-4o effectiveness to identify the types of errors made by the student modeled through GPT-3.5-turbo, Llama-3-8B-Instruct, and Mistral-7B-Instruct-v0.3 while solving math exercises based on the temperature parameter; 2) the investigation of the extent to which LLM teachers can generate pedagogically relevant hints i.e. hints that do not provide the answers to simulated students and whether the temperature setting can influence the ability of these simulated student models to self-correct after receiving such hints; 3) the design of several types of prompts, grouped into two categories: specialized prompts and general prompts, used to prompt the teacher model to generate hints – these two categories differ in their approach: specialized prompts are designed to correct a specific aspect by taking into account the initial answer given by the simulated student, while the general prompts provide hints based on common mistakes

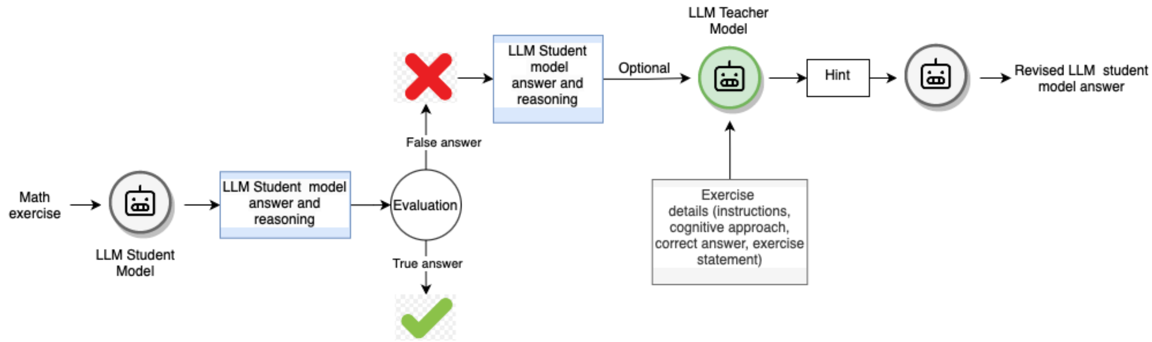


Figure 1: General approach: An LLM acting as a student solves a math exercise and provides its answer and reasoning. If the answer is incorrect, the exercise details (instructions, cognitive approach, correct answer, exercise statement) are passed, with or without the LLM student's answer and reasoning (depending on the hint-generation prompt type), to another LLM acting as a teacher. The LLM teacher model generates a hint in the form of question using different hint-generation prompts. The hint is then provided to the same LLM student model to revise its response.

that the student model might make when solving math exercises, without considering the simulated student's initial response; 4) the comparison of these prompts to determine the most effective ones; 5) the evaluation of Llama-3-8B-Instruct as a teacher to generate hints for the GPT-3.5-turbo and Mistral-7B-Instruct-v0.3 students, using the effective prompts identified.

2. Related work

2.1. Advances in ITS in Education through LLMs

Thanks to their ability to provide students with a personalized and effective learning experience, ITSs have gained popularity in the field of education (Winkler and Söllner, 2018). According to (Feng et al., 2021), these systems can be classified into four main categories. **Dialogue-based tutoring ITS**, such as AutoTutor (Graesser et al., 2004) and Beetle (Dzikovska et al., 2010) leverage natural language to identify students' misconceptions and respond to their prompts. **Constraint-based scaffolding models** (Mitrovic et al., 2013), exemplified by KERMIT (Suraweera and Mitrovic, 2002) use constraints predefined by human experts to respond to student queries. **Model tracing** (Liu et al., 2022; Sonkar et al., 2020) monitors students' knowledge states to capture their problem-solving skills. **Bayesian network modeling** (Corbett and Anderson, 1995) extends model tracing by using Bayesian networks.

Furthermore, recent advances in generative artificial intelligence, particularly with the emergence of LLMs such as GPT-4 (Bubeck et al., 2023) from OpenAI and more compact models like Llama (Touvron et al., 2023b) from Meta, have demonstrated their potential to significantly enhance these educational technologies. Their remarkable capabilities in generating human-like text and understanding complex linguistic patterns make them particularly well-suited for creating ITS that can interact with students in a more natural and interactive manner. For example, with Quizbot, (Ruan et al., 2019) showed the impact of advancements in LLMs on the evolution of educational chatbots. Additionally, (Roller et al., 2021) proposed a framework to further develop open-domain chatbots. Recently, GPT-4 integrations have been implemented in educational platforms such as Khan Academy's Khanmigo (noa, b) and Quizlet's Q-Chat (noa, c), demonstrating the effectiveness of these systems powered by OpenAI's GPT models. However, most ITSs based on LLMs simply use APIs of these models with a prompt-based strategy, which can limit their scalability while imposing high costs and access restrictions. This is why (Sonkar et al., 2023b) designed a framework called Conversational Learning with Analytical Step-by-Step Strategies (CLASS), aimed at creating ITS powered by performant LLMs, capable of assisting students by posing step-by-step questions. They also introduced a proof-of-concept ITS, called SPOCK, which is trained using the CLASS framework, with a focus on introductory-level college biology content. This framework uses two datasets (Scaffolding dataset and Conversational dataset) generated by GPT-4, which are used to train the SPOCK model (Vicuna 13B) through supervised fine-tuning (SFT). Although some studies address the generation of hints, they primarily focus on the field of biology (Sonkar et al., 2023b), which is not our area of study. These works are often specific to their domain and are not easily generalizable to other disciplines, such as mathematics. This study to our knowledge, is original as it studies and evaluates the use of LLMs for question hint generation in mathematics, thereby filling a gap in the current literature.

2.2. LLMs for Feedback and Hint Generation

Before the recent advances in generative artificial intelligence, one of the commonly used approaches for generating feedback or hints in the educational field relied on features designed to detect errors in students' responses. A rule-based system was then employed to provide relevant comments or hints (Botelho et al., 2023; Kochmar et al., 2020; Lan et al., 2015; Razzaq et al., 2020; Singh et al.; Song et al., 2021). This approach was popular due to its interpretability and reliability. However, it required significant human effort to adapt to new types of questions. With the advent of LLMs, a more general approach for generating feedback or hints involves using these advanced models either through prompting (Al-Hossami et al., 2023; McNichols et al., 2024; Nguyen et al., 2023; Steiss et al., 2023; Wang et al., 2024) or fine-tuning (Qinjin Jia et al., 2022). Several studies have been conducted in this area, particularly in the context of programming education. For example, (Roest et al., 2023) explored how LLMs can contribute to programming education by providing students with automated hints for the next steps. They found that most of the feedback messages generated by LLMs describe a specific step to follow and are personalized based on the student's code and approach. However, these hints can sometimes contain misleading information and lack sufficient detail when students are nearing the end of an exercise.

Similarly, (Phung et al., 2023) studied the role of generative AI models in providing human tutor-like hints to help students resolve errors in their faulty programs. However, when prompting pre-trained LLMs, it is crucial that these models exhibit good behavior and a clear understanding of educational objectives. For example, despite these advances, a major challenge these models face is their limited accuracy in handling mathematical calculations. GPT-4, for instance, showed only 59% accuracy on basic tasks like three-digit multiplication (Dziri et al., 2023). To enhance the mathematical capabilities of LLMs, several methods have been developed, such as the evol-instruct framework of WizardMath (Luo et al., 2023), combining LLMs with symbolic solvers (He-Yueya et al., 2023), or the introduction of "code soliloquies" by (Sonkar et al., 2023a), which allow for precise invocation of Python calculations whenever a student's response requires it. Nevertheless, the hints or feedback generated by the models should be clear, simple, encouraging, positive in tone, and relevant to the learning objectives (Jangra et al., 2024) as well as address the individual needs of learners (SUAIB, 2019; noa, d). Many research efforts concentrate on the generation of feedback, which is different from hint generation. These studies are predominantly oriented towards the field of programming, which is distinct from the focus of our paper. Additionally, works in mathematics on feedback or hint generation often benefit from pre-existing datasets, consisting of feedback or hint generated by human experts (Scarlatos et al., 2024). This makes the task of generating feedback somewhat more approachable. In contrast, our study is the first to attempt generating hints in mathematics using LLMs without the advantage of a pre-existing dataset created by human experts. The absence of such reference data makes our approach innovative and underscores the importance of this research, which aims to address this gap in future studies.

2.3. Simulation of Human Behaviors by LLMs

LLMs have the ability to simulate human behaviors, a capability that has shown promising applications in the field of education. For instance, (Markel et al., 2023) introduced GPTeach, an interactive teacher training tool based on a chat system. This tool allows novice teachers to practice with simulated students, using GPT to take a prompt and generate a response to it (Brown et al., 2020). In our study, inspired by the work presented in (Markel et al., 2023), we simulate both teachers and students. Moreover, other studies have suggested that LLMs can be prompted to replicate desired model behaviors (Jiang et al., 2021, 2022; Liu et al., 2021). Recently, a study (Argyle et al., 2023) demonstrated that with specific prompting techniques, LLMs can successfully simulate human sub-populations. This work is supported by (Arora et al., 2022), who described various effective prompting techniques. In contrast to these approaches, (Park et al., 2023, 2022) used specialized GPT prompting techniques to simulate not just one person, but an entire online community composed of simulated individuals, each with a unique personality.

3. Methodology

As a first step, generating high-quality hints requires a rigorous approach to understanding how hints are created when LLMs play the roles of students and teachers. Then, how can we simulate a student and a teacher with LLMs? To achieve this, we used prompt engineering (Sahoo et al., 2024).

3.1. Prompt design for question hint generation

To obtain high-quality hints, various approaches were considered for designing prompts and compare them. This led to the development of specific pipelines allowing to test and identify the best prompts for generating relevant and useful hints. A detailed picture of the pipeline can be found in Appendix $4(a)$. This pipeline is divided into two main stages. The first stage is executed first, followed by the second stage using the datasets obtained from the first stage.

The first stage implements the generation and classification of the student answers. The objective of this step is to create, for each exercise, a diverse dataset of incorrect answers, including different incorrect reasoning or solutions.

1. *Resolution by the student model*: For each exercise, the student model solves it using the prompt described in Appendix A.3.1, producing reasoning and a response.

2. *Verification of the response by GPT-4o*: The response from the student model is compared to the correct solution using the prompt in Appendix A.3.4.
   These two steps are repeated a predetermined number of times *num_simulations* , where *num_simulations* represents the number of attempts to solve the same exercise by the student model.

3. *Error classification*: After manually evaluating the incorrect responses of simulated students, we identified the following common types of errors: misunderstanding, interpretation, calculation, simplification, algebraic errors, partial answers, term grouping, and incorrect substitution. To automate the determination of these error types, we employed GPT-4o using few-shot prompting (Sahoo et al., 2024), as detailed in the prompt described in Appendix A.3.4, which includes examples for each common error type. GPT-4o was then used to categorize error types or groups of errors (multiple errors present in a simulated student's incorrect response), allowing for the creation of a diverse error dataset with various reasoning mistakes or incorrect solutions, using the prompt described in Appendix A.3.5.

At the end, a dataset of exercise solutions is produced that includes all the student model's responses, whether correct or incorrect, corresponding to the number of simulations conducted. We also have an error classification dataset containing various incorrect responses and reasoning by error type.

The second stage implements the generation of hints and revision of various incorrect responses. Only one incorrect answer per error type is retained from the error classification dataset obtained in stage 1 to form the one used in this step.

1. *Hint generation by the teacher Model*: For an incorrect response in the dataset, the teacher model generates appropriate hints based on the type of prompt used for hint generation. These prompts are described in Appendix A.4.

2. *Revision of the response by the student model*: The student model uses these hints to revise its initial response using the prompt in the Appendix A.3.2.

3. *Verification of the revised response by GPT-4o*: The revised response is checked again by GPT-4o to ensure it is now correct. This is done by using the prompt described

in Appendix A.3.3.

These steps are repeated for each incorrect response in the dataset, as well as for each type of hint generation prompt. The goal is to find the best prompts, i.e., those that allow the best correction of errors by the student models.

The models simulating the students are powered by GPT-3.5-turbo, Llama-3-8B-Instruct, and Mistral-7B-Instruct-v0.3, while those playing the role of the teachers are either GPT-4o or Llama-3-8B-Instruct. GPT-4o was primarily used as the teacher and for intermediate steps in the pipeline, except for the student model stages, because it is currently among the most powerful model available. As mentioned earlier, we defined several prompts. The resolution of exercises and the revision of answers by the student models were done through zero-shot prompting (Sahoo et al., 2024), meaning without providing examples. We chose to use zero-shot prompting because, when using zero-shot Chain of Thought (CoT) (Jin et al., 2024), the student model made fewer errors, which was not relevant to our study. Indeed, we were looking for a model that produced a balanced mix of erroneous and correct results The phases of answer verification and error classification are also performed by the GPT-4o model in zero-shot prompting. For hint generation, the prompts were also written in zero-shot prompting. Once we selected the best prompts, a second pipeline was created to evaluate these prompts.

3.2. Pipeline for evaluating the best prompts

To evaluate these prompts, we implemented the pipeline illustrated in Figure 4(b). This pipeline describes a process for evaluating the best prompts. Initially, responses are collected from a dataset of exercise resolutions generated by a student model. If a response is incorrect, a teacher model provides a hint using the best-selected prompt. The student model then revises its response based on this hint. The revised response is subsequently submitted to GPT-4o for re-verification, and the outcome is recorded. This process is repeated for all responses in the dataset. By the end of the pipeline, we have a comprehensive dataset containing all responses before and after hint generation, along with their corresponding evaluations.

4. Experiments settings

For these experiments, we worked with the mathematics exercises from MIA Seconde educational software. MIA Seconde is an educational tool developed by EvidenceB, which offers remediation exercises in French and mathematics for students in general, technological, and vocational high school classes. These exercises were initially developed through collaboration with researchers in cognitive science and neuroscience, drawing on insights into how the student's brain functions and theories about human mathematical cognition. The theory underlying the MIA Seconde exercises is documented in guides called pedagogical summaries. A pedagogical summary is an official document from EvidenceB that serves as a reference for the design of these exercises. These summaries are organized into different modules, objectives, and activities, each focusing on a specific knowledge or skill.

Modules correspond to an overarching skill developed in the exercises. It generally refers to a sub-discipline of French or mathematics, for example, "quantities and measurements".

Objectives concisely specifies the skills that will be tested. It describes what students should master after practicing the provided exercises, for example, "calculating areas and volumes, and performing unit conversions". Activities clarify the theme of the exercise. It is characterized by a detailed description of the researchers' cognitive approach and a typical explained exercise. The activity often corresponds to a certain level of difficulty within an objective.

We worked with four different exercises: two exercises for the same module, and two others corresponding to two distinct modules. Each exercise includes its statement, the instructions to follow, the cognitive process, as well as the solution. They are written in French. More informations about details on the implementation and the exercises along with their associated pedagogical elements can be found respectively in Appendix A.2 and A.6.

5. Experiments and results

In this section, we analyze the performance of GPT-4o and Llama-3-8B-Instruct as teachers by evaluating their ability to generate helpful hints for solving math problems. We examine student model errors, the best hint generation prompts, and the impact of temperature on the models' problem-solving and revision skills.

5.1. What types of errors student models make when solving math exercises, and how does it depend on the temperature parameter?

The types of errors considered are: Comprehension Error: the student does not understand the problem or instructions clearly; Partial Response: the student provides only part of the answer and fails to complete it correctly; Term Grouping Error: the student incorrectly combines or groups terms in an expression; Simplification Error: the student simplifies an expression incorrectly; Calculation Error: the student performs mathematical operations incorrectly; Incorrect Substitution Error: the student substitutes the wrong value in an expression or equation; Interpretation Error: the student misinterprets the instructions or data; Algebraic Error: the student makes mistakes in algebraic manipulations. These errors were identified through manual evaluation and common student mistakes in math, then used in *"few-shot prompting"* with GPT-4o for evaluating student model responses, as mentioned in the error classification phase of the pipeline (see 3.1). We manually verified the phases of answer checking, error classification and error type determination to ensure GPT-4o's was not making any mistakes. In most cases, the results were correct (around 98% of the time).

To analyze the types of errors made by the student models when solving the four exercises under standard settings (default temperature), the first two steps of the initial stage in the pipeline for determining the best prompt for hint generation (see Figure $4(a)$) were executed 40 times for each exercise. It is equally interesting to observe how the results vary when the temperature value is adjusted, as temperature is a parameter that controls the creativity and diversity of the responses generated by the model. To explore this, the process was repeated for each temperature value (0, 0.2, 0.5, 0.8, 1) across all models. Studying the effects of different temperature values would also allow us to determine whether the temperature parameter can influence the student model's ability to incorporate a hint during revision process. The verification, evaluation, error detection, and classification steps were performed by GPT-4o, as mentioned earlier, with a temperature value set to 0 to ensure accurate results. It was observed that at higher temperatures (> 0.2), these steps were not reliable.

Studying the effects of different temperature values would also allow us to determine whether the temperature parameter can influence the student model's ability to incorporate a hint during the revision process.

Table 1: A checkmark (✓) indicates the presence of a specific type of error at a given temperature, while a dash (—) indicates its absence. The table shows errors made by GPT-3.5-turbo (G3.5), Llama-3-8B-Instruct (L8B), and Mistral-7B-Instruct-v0.3 (M7B) in exercise 1 (module 1). As temperature increases, the number and variety of errors tend to rise, varying across models.

Error Type	Temp 0.2			Temp 0.5			Temp 0.8			Temp 1.0		
	G3.5	L8B	M7B	G3.5	L8B	M7B	G3.5	L8B	M7B	G3.5	L8B	M7B
Comprehension Error, Grouping of Terms Error	—	—	—	—	—	—	—	-	—	—	—	—
Interpretation Error, Calculation Error	—	—	—	—	—	—	—	—	—	✓	—	—
Comprehension Error, Calculation Error	—	—	—	—	—	—	✓	—	✓	—	—	✓
Comprehension Error	—	—	✓	✓	—	✓	✓	✓	✓	✓	✓	✓
Calculation Error, Interpretation Error	—	—	—	—	—	—	—	—	—	✓	—	—
Interpretation Error	✓	✓	✓	✓	✓	—	✓	✓	✓	✓	✓	—
Grouping of Terms Error	—	—	—	—	—	—	—	—	—	—	-	—
Calculation Error	✓	—	—	—	—	—	✓	—	—	—	✓	—
Simplification Error	—	—	—	✓	—	—	—	—	—	✓	✓	—
Comprehension Error, Interpretation Error	✓	—	✓	✓	—	✓	—	✓	✓	✓	✓	✓
partial response	—	—	—	✓	—	—	✓	—	—	✓	—	—
Simplification Error, Comprehension Error	—	—	—	—	—	—	✓	—	—	✓	—	—
Comprehension Error, Calculation Error, Interpretation Error	—	—	—	—	—	—	—	—	✓	—	—	—

We observed that the types of errors made by the student models vary significantly depending on the exercise, the model used, and the temperature value applied. Indeed, the higher the temperature value, the more likely the models are to make errors, as shown in Table 1. This Table summarizes Figure 10, offering a concise view of the types of errors encountered at each temperature and for each model in exercise 1 (module 1). In this exercise, Llama made 5 errors, Mistral made 5, and GPT made 9 types of errors. So, Llama-3 and Mistral models tend to make fewer errors types than the GPT-3.5-turbo model. It is also worth mentioning that the Mistral model exhibits a relatively high number of decoding errors compared to the other two. For more details, you can refer to the Appendix A.5.3, where we present the evolution of error types based on temperature for other exercises.

5.2. What type of prompt is most effective for generating hints with GPT-4o?

In order to select the best prompts for generating hints, we defined several prompts, which can be grouped into two categories.

First, **prompts based on the types of errors made by the student model** aim to correct a key aspect of the student's response. They are based on reasoning, the student's response, the instruction, the cognitive approach, the correct answer, and the exercise, and incorporate these elements into the context. They are labelled as follow: prompt_hint_reason is based on the student's reasoning; prompt_hint_method is based on the method used by the student; prompt_hint_concpt is based on the application of concepts;

prompt_hint_calcul is based on calculations; prompt_hint_interp is based on problem interpretation; prompt_hint_all combines all the above aspects (calculations, reasoning, interpretation, concepts, method); prompt_hint_part_res is used to complete partial answers.The prompts details are presented in Appendix A.4.2

Second, **general or baseline prompts** do not consider the student's reasoning or response, and do not incorporate these elements into the context. They are based only on the exercise, instruction, correct answer, and cognitive approach, incorporating these into the context. They are labelled prompt_baseline_one which corresponds to BaselineOne and provides progressive hints to assist the student model and prompt_baseline_two which correspond to BaselineTwo prompt which provides hints to assist the student model, based solely on common mistakes that students might make in mathematics. The prompts details are presented in Appendix A.4

These hints must meet the following criteria: be in the form of a question, not include the correct answer or a part of it, follow the cognitive approach, be relevant to the exercise. These criteria were introduced into the hint generation prompts to guide the LLM in adhering to these conditions.

The second stage of the pipeline $4(a)$ is then executed by repeating it 10 times for each of the 9 types of prompts, each temperature value, and each incorrect response from the error classification dataset obtained in stage 1 of this pipeline. Note that when revising the student's response, the same temperature value used during the resolution is applied. For the teacher model, we used a temperature of 1 in all experiments to obtain diverse hints. The mean revision error rate compares prompts by averaging the error revision rates for all encountered errors at a given temperature. For each error type, the error revision rate is calculated as 1 minus the ratio of correct responses to the total number of responses (correct and incorrect) from 10 repetitions.

Table 2: Mean revision error rate for each prompt, model, and temperature on exercise 1(module 1). Lower mean revision error rates indicate more effective prompts.

Prompt	Temp 0.2			Temp 0.5			Temp 0.8			Temp 1.0		
	G3.5	L8B	M7B	G3.5	L8B	M7B	G3.5	L8B	M7B	G3.5	L8B	M7B
prompt_hint_reason	0.20	0.80	0.37	0.34	0.80	0.42	0.42	0.63	0.31	0.51	0.60	0.32
prompt_hint_method	0.27	0.80	0.53	0.46	0.70	0.13	0.48	0.70	0.40	0.54	0.63	0.54
prompt_hint_concpt	0.23	0.80	0.41	0.48	0.80	0.31	0.48	0.50	0.40	0.44	0.70	0.36
prompt_hint_calcul	0.27	0.90	0.44	0.44	0.90	0.38	0.50	0.70	0.53	0.43	0.55	0.11
prompt_hint_interp	0.20	0.60	0.37	0.40	0.90	0.30	0.42	0.73	0.39	0.44	0.65	0.78
prompt_hint_all	0.27	0.70	0.53	0.44	0.90	0.29	0.40	0.70	0.35	0.50	0.60	0.47
prompt_hint_part_res	0.27	0.80	0.37	0.48	0.60	0.29	0.44	0.77	0.54	0.53	0.55	0.31
prompt_baseline_one	0.08	1.00	0.60	0.30	0.71	0.00	0.67	0.71	0.49	0.44	0.63	0.67
prompt_baseline_two	0.20	1.00	0.73	0.48	0.71	0.00	0.33	0.47	0.25	0.33	0.90	0.33

The different prompts enable the learning models to correct their errors, as shown in Table 2, the lower the mean revision error, the more effective the prompt is. Indeed, regardless of the temperature used, the cues generated by GPT-4o through these prompts allow the LLM models to correct their responses, sometimes entirely. For example, in the case of the Mistral model at a temperature of 0.5, the mean revision error rate is 0 for the baseline-type prompts, which means that all errors where correctly revised. Similarly, for GPT-3.5-turbo, the BaselineTwo prompt has a mean revision error of 0, indicating that all the cues generated by these prompts enabled GPT-3.5-turbo or Mistral to correct their initial response. Table 2 is a condense version of the Figure 8 in Appendix A.5.2 where more details about these results are described.

We consider the best prompt to be the one that enables the student model to correct itself the most times over the 10 repetitions. This prompt is identified by selecting the one with the lowest mean revision error rate for each temperature across all exercises. The top prompts are those that appear most frequently as the best. The best specialized prompt was found to be the ***one based on calculation errors***, while the best baseline prompt is ***BaselineTwo***. These two prompts were therefore used in the continuation of our experiments.

5.3. What is the influence of the temperature parameter on the performance of the student models in solving exercises and revising answers?

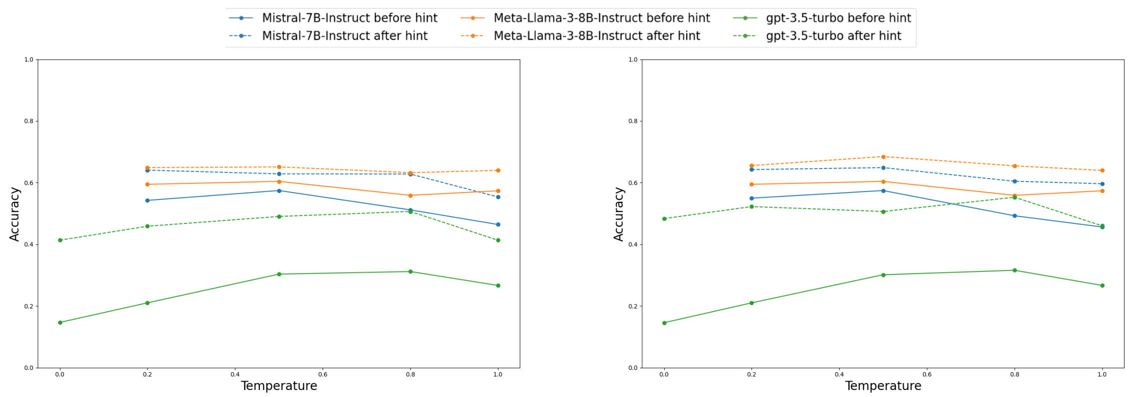
We worked with the best prompts from both categories, specifically the calculation-based prompt for the specialized prompts and BaselineTwo for the baseline-type prompts. To study how temperature could impact the resolution and revision by the student models, we used the validation pipeline shown in Figure $4(b)$. Note that when revising the student's response, the same temperature value used during the resolution is applied. Metric such as accuracy was used to quantify the performance of these models using the best prompts. It was computed as the number of correct responses out of the 40 repetitions divided by the number of responses (number of correct+number of incorrect responses).

Figures 3 and 2 do not show a clear direct link between the ability of student models to solve exercises and revise their answers when GPT-4o or Llama-3-8B-instruct are used as teachers. However, for the GPT student model, we observe that accuracy during both solving and revision decreases when the temperature is set to 1, which is not always the case for the other student models. Conversely, for lower temperatures (e.g., 0 or 0.2), accuracy increases.

5.4. How does the accuracy of student model problem-solving evolve before and after providing hints when they are guided by Llama-3-8B-Instruct compared to GPT-4o?

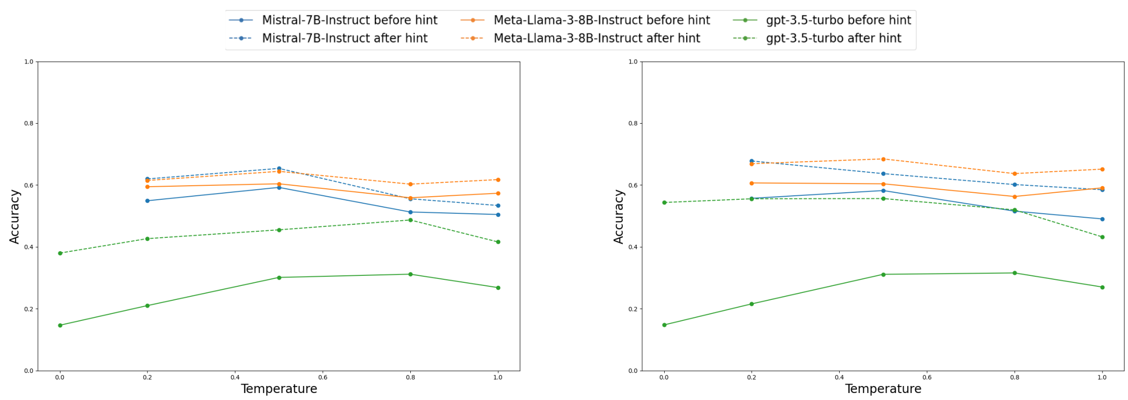
Since our goal is to use LLMs for hint generation, we were curious to see how a smaller language model like Llama-3-8B-Instruct would perform in generating hints. Therefore, we used it as the teacher model in the pipeline shown in Figure $4(b)$, utilizing the best prompts.

Figures 3 and 2 show that whether the BaselineTwo prompt or the one based on calculation errors is used with GPT-4o or Llama-3-8B-instruct, the models manage to correct themselves. A notable improvement is particularly observed for the GPT-3.5-turbo model. Indeed, the accuracy of this model increases significantly after receiving a hint, even if its initial accuracy was low. In contrast, the other models show a more moderate increase. When GPT-4o is used as the teacher, the hints provided by the error-based prompt seem more effective in improving the student models' performance than those from the BaselineTwo prompt. However, the opposite effect is observed with Llama-3-8B-instruct as the teacher. Comparing the two teacher models, the figures suggest that the overall performance is better



(a) GPT-4o (b) Llama-3-8B-instruct

Figure 2: Comparison of accuracy before and after providing hints across four exercises for different student models, using GPT-4o and Llama-3-8B-instruct as teacher models with **the best specialized hint generation prompt focused on calculation errors**. The results show improved performance when using Llama-3-8B-instruct as the teacher model.



(a) GPT-4o (b) Llama-3-8B-instruct

Figure 3: Comparison of accuracy before and after providing hints across four exercises for different student models, using GPT-4o and Llama-3-8B-instruct as teacher models with **the best baseline-type hint generation prompt, named BaselineTwo**. The results show improved performance when using Llama-3-8B-instruct as the teacher model.

with Llama-3-8B-instruct. However, although accuracy is higher with Llama-3-8B-instruct, it would be crucial to verify the quality and relevance of the generated hints.

For more detailed results by exercise, you can refer to section A.5.1. There, we show how the accuracy evolves for each exercise. However, exercise 3 (module 7) is particularly challenging for the models to solve. Only the GPT student model manages to correct itself after being given a hint, whether the teacher is GPT-4o or Llama-3-8B-instruct.

6. Discussion and Limitations

Our work addresses a gap in hint generation research within the field of mathematics education. We demonstrated that language models could identify their own errors when acting as students, with error patterns varying based on the temperature setting. Higher temperatures led to more diverse outputs but increased errors, while lower temperatures produced more deterministic. This error detection was crucial for selecting effective prompts for generating a synthetic hint dataset. We found that prompts focused on error correction and the BaselineTwo prompt were most successful. This aligns with the known challenges language models face with calculations and reasoning tasks. Interestingly, our results differ from a previous study (Renze and Guven, 2024) on Multiple-Choice Question Answering (MCQA), which found that temperature did not impact problem-solving abilities but affected text variability. In contrast, we observed no clear link between temperature and problem-solving in our non-MCQA tasks, where we used only zero-shot approaches.

GPT-3.5-turbo showed the most effective self-correction after receiving optimized hints, likely due to the prompts being tailored for base GPT models, leading to a significant correction gap compared to other student models. Mistral-7B-instruct-v3 and Llama-3-8B-instruct, however, already had high accuracy with the hints, making further improvement harder, though their additional corrections remain noteworthy.

Notably, the Llama-3-8B-instruct model outperformed GPT-4o in accuracy when using the best prompts, challenging the assumption that larger models like GPT-4o are always superior. Future work should include a qualitative analysis of the generated hints in relation to pedagogical criteria and their relevance, as well as explore the potential of fine-tuning smaller models, such as Llama-3-8B-instruct, for hint generation.

This study has, however, several limitations. First, we only used four exercises from different modules, which is not sufficient for a comprehensive analysis, even though each exercise was solved 40 times. Results may differ with other exercise variants within these modules. Additionally, the cost of the API limited the number of exercises we could analyze.

The prompts for hint generation were optimized for GPT models, not for other models, so more tailored prompts might produce better results for non-GPT models. We also limited our analysis to GPT-4o and GPT-3.5-turbo due to cost constraints. Error type classification was evaluated only with GPT-4o, without the involvement of human experts, though some human verification was done. Including expert evaluation would provide deeper insights. Finally, the lack of qualitative analysis of the generated hints is another limitation, as such an analysis could offer valuable context and improve the overall assessment of hint quality.

Acknowledgments

We thank the anonymous reviewers for their constructive feedback. We also thanks the entire team at EvidenceB for hosting this project.

References

Introducing ChatGPT, a. URL https://openai.com/index/chatgpt/.

Meet Khanmigo: Khan Academy's AI-powered teaching assistant & tutor, b. URL https://www.khanacademy.org/khan-labs.

Q-Chat: Meet your new AI tutor | Quizlet, c. URL https://quizlet.com/qchat-personal-ai-tutor.

Recognizing and Responding to Cultural Differences in the Education of Culturally and Linguistically Diverse Learners - Steven P. Chamberlain, 2005, d. URL https://journals.sagepub.com/doi/abs/10.1177/10534512050400040101.

Erfan Al-Hossami, Razvan Bunescu, Ryan Teehan, Laurel Powell, Khyati Mahajan, and Mohsen Dorodchi. Socratic Questioning of Novice Debuggers: A Benchmark Dataset and Preliminary Evaluations. In Ekaterina Kochmar, Jill Burstein, Andrea Horbach, Ronja Laarmann-Quante, Nitin Madnani, Anaïs Tack, Victoria Yaneva, Zheng Yuan, and Torsten Zesch, editors, *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*, pages 709–726, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.bea-1.57. URL https://aclanthology.org/2023.bea-1.57.

Lisa P. Argyle, Ethan C. Busby, Nancy Fulda, Joshua Gubler, Christopher Rytting, and David Wingate. Out of One, Many: Using Language Models to Simulate Human Samples. *Political Analysis*, 31(3):337–351, July 2023. ISSN 1047-1987, 1476-4989. doi: 10.1017/pan.2023.2. URL http://arxiv.org/abs/2209.06899. arXiv:2209.06899 [cs].

Simran Arora, Avanika Narayan, Mayee F. Chen, Laurel Orr, Neel Guha, Kush Bhatia, Ines Chami, Frederic Sala, and Christopher Ré. Ask Me Anything: A simple strategy for prompting language models, November 2022. URL http://arxiv.org/abs/2210.02441. arXiv:2210.02441 [cs].

Anthony Botelho, Sami Baral, John Erickson, Priyanka Benachamardi, and Neil Heffernan. Leveraging natural language processing to support automated assessment and feedback for student open responses in mathematics. *Journal of Computer Assisted Learning*, 39: n/a–n/a, February 2023. doi: 10.1111/jcal.12793.

Mark Bray. *The Shadow education system: private tutoring and its implications for planners*. January 1999. Journal Abbreviation: http://lst-iiep.iiep-unesco.org/cgi-bin/wwwi32.exe/[in=epidoc1.in]/?t2000=026377/(100) Publication Title: http://lst-iiep.iiep-unesco.org/cgi-bin/wwwi32.exe/[in=epidoc1.in]/?t2000=026377/(100).

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners, July 2020. URL http://arxiv.org/abs/2005.14165. arXiv:2005.14165 [cs].

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. Sparks of Artificial General Intelligence: Early experiments with GPT-4, April 2023. URL http://arxiv.org/abs/2303.12712. arXiv:2303.12712 [cs].

Albert T. Corbett and John R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modelling and User-Adapted Interaction*, 4(4):253–278, 1995. ISSN 0924-1868, 1573-1391. doi: 10.1007/BF01099821. URL http://link.springer.com/10.1007/BF01099821.

Myroslava O. Dzikovska, Johanna D. Moore, Natalie Steinhauser, Gwendolyn Campbell, Elaine Farrow, and Charles B. Callaway. Beetle II: A System for Tutoring and Computational Linguistics Experimentation. In Sandra Kübler, editor, *Proceedings of the ACL 2010 System Demonstrations*, pages 13–18, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL https://aclanthology.org/P10-4003.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena D. Hwang, Soumya Sanyal, Sean Welleck, Xiang Ren, Allyson Ettinger, Zaid Harchaoui, and Yejin Choi. Faith and Fate: Limits of Transformers on Compositionality, October 2023. URL http://arxiv.org/abs/2305.18654. arXiv:2305.18654 [cs].

Shi Feng, Alejandra Magana, and Dominic Kao. A Systematic Review of Literature on the Effectiveness of Intelligent Tutoring Systems in STEM. pages 1–9, October 2021. doi: 10.1109/FIE49875.2021.9637240.

Arthur Graesser, Shulan Lu, G. Jackson, Heather Mitchell, Mathew Ventura, Andrew Olney, and Max Louwerse. AutoTutor: a Tutor with Dialogue in Natural Language. *Behavior Research Methods*, 36:180–192, June 2004. doi: 10.3758/BF03195563.

Arthur Graesser, Mark Conley, and Andrew Olney. Intelligent tutoring systems. *APA educational psychology handbook: Vol. 3. Applications to learning and teaching*, 3:451–473, January 2012. ISSN 1-4338-0999-0. doi: 10.1037/13275-018.

Hippolyte Gros, Jean-Pierre Thibaut, and Emmanuel Sander. Semantic congruence in arithmetic: A new conceptual model for word problem solving. *Educational Psychologist*, 55 (2):69–87, April 2020. ISSN 0046-1520, 1532-6985. doi: 10.1080/00461520.2019.1691004. URL https://www.tandfonline.com/doi/full/10.1080/00461520.2019.1691004.

Joy He-Yueya, Gabriel Poesia, Rose E. Wang, and Noah D. Goodman. Solving Math Word Problems by Combining Language Models With Symbolic Solvers, April 2023. URL http://arxiv.org/abs/2304.09102. arXiv:2304.09102 [cs].

Anubhav Jangra, Jamshid Mozafari, Adam Jatowt, and Smaranda Muresan. Navigating the Landscape of Hint Generation Research: From the Past to the Future, April 2024. URL http://arxiv.org/abs/2404.04728. arXiv:2404.04728 [cs].

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

Ellen Jiang, Edwin Toh, Alejandra Molina, Aaron Donsbach, Carrie Cai, and Michael Terry. GenLine and GenForm: Two Tools for Interacting with Generative Language Models in a Code Editor. pages 145–147, October 2021. doi: 10.1145/3474349.3480209.

Ellen Jiang, Kristen Olson, Edwin Toh, Alejandra Molina, Aaron Donsbach, Michael Terry, and Carrie J Cai. PromptMaker: Prompt-based Prototyping with Large Language Models. In *CHI Conference on Human Factors in Computing Systems Extended Abstracts*, pages 1–8, New Orleans LA USA, April 2022. ACM. ISBN 978-1-4503-9156-6. doi: 10.1145/3491101.3503564. URL https://dl.acm.org/doi/10.1145/3491101.3503564.

Feihu Jin, Yifan Liu, and Ying Tan. Zero-Shot Chain-of-Thought Reasoning Guided by Evolutionary Algorithms in Large Language Models, February 2024. URL http://arxiv.org/abs/2402.05376. arXiv:2402.05376 [cs].

John Kinnebrew, James Segedy, and Gautam Biswas. Integrating Model-Driven and Data-Driven Techniques for Analyzing Learning Behaviors in Open-Ended Learning Environments. *IEEE Transactions on Learning Technologies*, 10:1–1, December 2015. doi: 10.1109/TLT.2015.2513387.

André Knops. *Neural and cognitive processes underlying numerosity perception and mental arithmetic*. PhD thesis, Université Paris Cité, 2022.

Ekaterina Kochmar, Dung Do Vu, Robert Belfer, Varun Gupta, Iulian Vlad Serban, and Joelle Pineau. Automated Personalized Feedback Improves Learning Gains in an Intelligent Tutoring System, May 2020. URL http://arxiv.org/abs/2005.02431. arXiv:2005.02431 [cs].

Andrew S. Lan, Divyanshu Vats, Andrew E. Waters, and Richard G. Baraniuk. Mathematical Language Processing: Automatic Grading and Feedback for Open Response Mathematical Questions, January 2015. URL http://arxiv.org/abs/1501.04346. arXiv:1501.04346 [cs, stat].

Naiming Liu, Zichao Wang, Richard Baraniuk, and Andrew Lan. Open-ended Knowledge Tracing for Computer Science Education. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural*

Language Processing, pages 3849–3862, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.254. URL https://aclanthology.org/2022.emnlp-main.254.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing, July 2021. URL http://arxiv.org/abs/2107.13586. arXiv:2107.13586 [cs].

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, and Dongmei Zhang. WizardMath: Empowering Mathematical Reasoning for Large Language Models via Reinforced Evol-Instruct, August 2023. URL http://arxiv.org/abs/2308.09583. arXiv:2308.09583 [cs].

Julia M. Markel, Steven G. Opferman, James A. Landay, and Chris Piech. GPTeach: Interactive TA Training with GPT-based Students. In *Proceedings of the Tenth ACM Conference on Learning @ Scale*, pages 226–236, Copenhagen Denmark, July 2023. ACM. ISBN 9798400700255. doi: 10.1145/3573051.3593393. URL https://dl.acm.org/doi/10.1145/3573051.3593393.

Hunter McNichols, Wanyong Feng, Jaewook Lee, Alexander Scarlatos, Digory Smith, Simon Woodhead, and Andrew Lan. Automated Distractor and Feedback Generation for Math Multiple-choice Questions via In-context Learning, January 2024. URL http://arxiv.org/abs/2308.03234. arXiv:2308.03234 [cs].

Antonija Mitrovic, Stellan Ohlsson, and Devon Barrow. The effect of positive feedback in a constraint-based intelligent tutoring system. *Computers & Education*, 60:264–272, January 2013. doi: 10.1016/j.compedu.2012.07.002.

Huy A. Nguyen, Hayden Stec, Xinying Hou, Sarah Di, and Bruce M. McLaren. Evaluating ChatGPT's Decimal Skills and Feedback Generation in a Digital Learning Game, June 2023. URL http://arxiv.org/abs/2306.16639. arXiv:2306.16639 [cs].

Allen Nie, Yash Chandak, Miroslav Suzara, Malika Ali, Juliette Woodrow, Matt Peng, Mehran Sahami, Emma Brunskill, and Chris Piech. The GPT Surprise: Offering Large Language Model Chat in a Massive Coding Class Reduced Engagement but Increased Adopters Exam Performances, April 2024. URL http://arxiv.org/abs/2407.09975. arXiv:2407.09975 [cs, stat].

Joon Sung Park, Lindsay Popowski, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Social Simulacra: Creating Populated Prototypes for Social Computing Systems, August 2022. URL http://arxiv.org/abs/2208.04024. arXiv:2208.04024 [cs].

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative Agents: Interactive Simulacra of Human Behavior, August 2023. URL http://arxiv.org/abs/2304.03442. arXiv:2304.03442 [cs].

Tung Phung, Victor-Alexandru Pădurean, Anjali Singh, Christopher Brooks, José Cambronero, Sumit Gulwani, Adish Singla, and Gustavo Soares. Automating Human Tutor-Style Programming Feedback: Leveraging GPT-4 Tutor Model for Hint Generation and GPT-3.5 Student Model for Hint Validation, December 2023. URL http://arxiv.org/abs/2310.03780. arXiv:2310.03780 [cs].

Qinjin Jia, Mitchell Young, Yunkai Xiao, Jialin Cui, Chengyuan Liu, Parvez Rashid, and Edward Gehringer. Insta-Reviewer: A Data-Driven Approach for Generating Instant Feedback on Students' Project Reports. July 2022. doi: 10.5281/ZENODO.6853099. URL https://zenodo.org/record/6853099. Publisher: Zenodo.

Renah Razzaq, Korinn Ostrow, and Neil Heffernan. Effect of Immediate Feedback on Math Achievement at the High School Level. pages 263–267. June 2020. ISBN 978-3-030-52239-1. doi: 10.1007/978-3-030-52240-7_48.

Matthew Renze and Erhan Guven. The Effect of Sampling Temperature on Problem Solving in Large Language Models, June 2024. URL http://arxiv.org/abs/2402.05201. arXiv:2402.05201 [cs].

Lianne Roest, Hieke Keuning, and Johan Jeuring. Next-Step Hint Generation for Introductory Programming Using Large Language Models, December 2023. URL http://arxiv.org/abs/2312.10055. arXiv:2312.10055 [cs].

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Eric Michael Smith, Y-Lan Boureau, and Jason Weston. Recipes for Building an Open-Domain Chatbot. In Paola Merlo, Jorg Tiedemann, and Reut Tsarfaty, editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 300–325, Online, April 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.24. URL https://aclanthology.org/2021.eacl-main.24.

Sherry Ruan, Liwei Jiang, Justin Xu, Bryce Joe-Kun Tham, Zhengneng Qiu, Yeshuang Zhu, Elizabeth L. Murnane, Emma Brunskill, and James A. Landay. QuizBot: A Dialogue-based Adaptive Learning System for Factual Knowledge. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–13, Glasgow Scotland Uk, May 2019. ACM. ISBN 978-1-4503-5970-2. doi: 10.1145/3290605.3300587. URL https://dl.acm.org/doi/10.1145/3290605.3300587.

Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications, February 2024. URL http://arxiv.org/abs/2402.07927. arXiv:2402.07927 [cs].

Alexander Scarlatos, Digory Smith, Simon Woodhead, and Andrew Lan. Improving the Validity of Automatically Generated Feedback via Reinforcement Learning, March 2024. URL http://arxiv.org/abs/2403.01304. arXiv:2403.01304 [cs] version: 1.

Rishabh Singh, Sumit Gulwani, and Armando Solar-Lezama. Automated Feedback Generation for Introductory Programming Assignments.

Dowon Song, Woosuk Lee, and Hakjoo Oh. Context-aware and data-driven feedback generation for programming assignments. pages 328–340, August 2021. doi: 10.1145/3468264.3468598.

Shashank Sonkar, Andrew E. Waters, Andrew S. Lan, Phillip J. Grimaldi, and Richard G. Baraniuk. qDKT: Question-centric Deep Knowledge Tracing, May 2020. URL http://arxiv.org/abs/2005.12442. arXiv:2005.12442 [cs, stat].

Shashank Sonkar, MyCo Le, Xinghe Chen, Naiming Liu, Debshila Basu Mallick, and Richard G. Baraniuk. Code Soliloquies for Accurate Calculations in Large Language Models, October 2023a. URL http://arxiv.org/abs/2309.12161. arXiv:2309.12161 [cs].

Shashank Sonkar, Naiming Liu, Debshila Basu Mallick, and Richard G. Baraniuk. CLASS: A Design Framework for building Intelligent Tutoring Systems based on Learning Science principles, October 2023b. URL http://arxiv.org/abs/2305.13272. arXiv:2305.13272 [cs].

Jacob Steiss, Tate, Steve Graham, Cruz, Hebert, Jiali Wang, Youngsun Moon, Tseng, and Warschauer. *Comparing the Quality of Human and ChatGPT Feedback on Students' Writing*. October 2023. doi: 10.35542/osf.io/ty3em.

RAHMAH SUAIB. THE USE OF VISUAL AUDITORY KINESTHETIC (VAK) LEARNING STYLES TO INCREASE STUDENTS' VOCABULARY. *DIDAKTIKA*, 11:239, June 2019. doi: 10.30863/didaktika.v11i2.169.

Pramuditha Suraweera and Antonija Mitrovic. KERMIT: A constraint-based tutor for database modeling. volume 2363, pages 377–387, June 2002. ISBN 978-3-540-43750-5. doi: 10.1007/3-540-47987-2_41.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and Efficient Foundation Language Models, February 2023a. URL http://arxiv.org/abs/2302.13971. arXiv:2302.13971 [cs].

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. Llama 2: Open Foundation and Fine-Tuned Chat Models, July 2023b. URL http://arxiv.org/abs/2307.09288. arXiv:2307.09288 [cs].

Rose E. Wang, Qingyang Zhang, Carly Robinson, Susanna Loeb, and Dorottya Demszky. Bridging the Novice-Expert Gap via Models of Decision-Making: A Case Study on Remediating Math Mistakes, April 2024. URL http://arxiv.org/abs/2310.10648. arXiv:2310.10648 [cs].

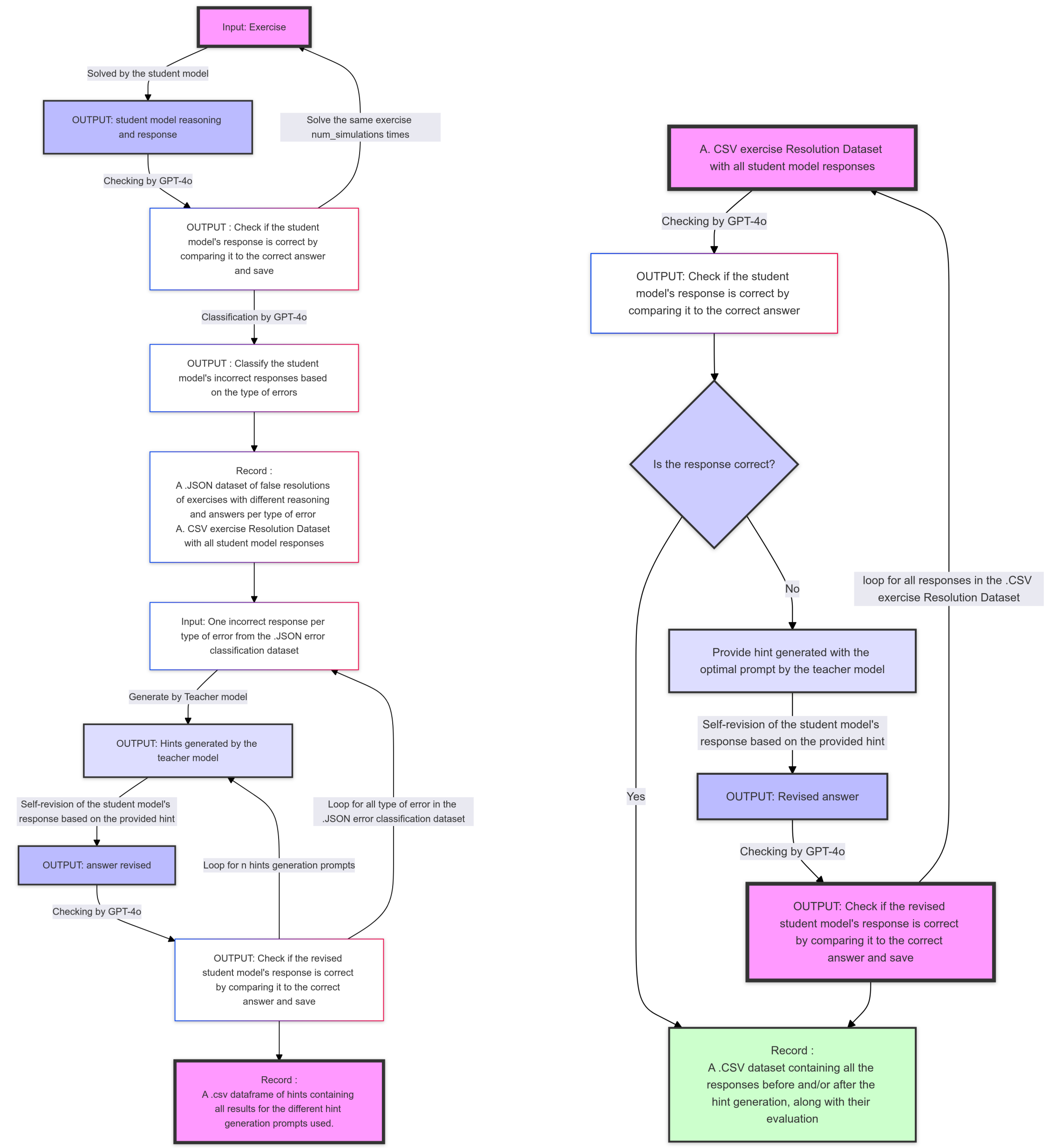
Rainer Winkler and Matthias Söllner. Unleashing the Potential of Chatbots in Education: A State-Of-The-Art Analysis. *Academy of Management Proceedings*, 2018:15903, April 2018. doi: 10.5465/AMBPP.2018.15903abstract.

Appendix A. Additional information

Contents

A.1. Experimental pipelines

This section outlines the pipelines used in our various experiments.



(a) Pipeline for determining the best prompt for hint generation

(b) Pipeline for evaluating the best prompt

A.2. Implementation details

We used the OpenAI API to interact with models based on GPT, such as GPT-4o and GPT-3.5-turbo. For other open-source models, like Llama-3-8B-Instruct and Mistral-7B-

Instruct-v0.3, we accessed the resources via the HuggingFace platform. Specifically for these models, the prompts were executed using a 512 GB setup with two A100 GPUs.

During the exercise resolution phase and the review of the student's responses across various pipelines, several questions arise:

How can we ensure that the hints are given to the same student who made the error or provided the response? How can we guarantee that no previous hint is reused for the same student in the current session?

The solution is to only add the student's initial solution to the context of the student for the current session. Indeed, whether using the API or open-source models, context management is done manually. There is no contextual dependency between requests unless we manually add the previous response to the context using the assistant role "role": "assistant", "content": "". By doing so, we ensure that the student who provided an incorrect response is the one who corrects their initial answer using the hints provided by the teacher.

A.3. Prompts

In this section, we present the various prompts used in the pipelines. These prompts were written in French in our experiments, but for the purposes of the paper, we have translated them into English.

A.3.1. Prompt for exercise resolution by the student model

```
{"role": "system", "content": "You are a high school student who must solve
mathematics exercises."},
{"role": "user", "content": '''Your objective is to answer the questions in the
exercises by following the given instructions.
Exercise and question: {exercise}
Instructions: {instruct}
Required answer format: use a JSON format with the following structure:
{"reasoning": "Explain your reasoning here...", "answer": "Provide your answer
here..."}
I emphasize that you must follow the required response format, and also that
you must answer the questions in the exercises by following the instructions
as given, without adding anything.'''}
```

A.3.2. Prompt for answer review by the student model

```
{"role": "system", "content": "You are a high school student who must solve mathematics
exercises."},
{"role": "user", "content": '''You provided an incorrect answer to a math exercise.
A teacher has given you a hint to help you understand your mistake and correct
it. Your objective is to review your response to the questions in the exercise
using the hint provided by the teacher.
Exercise and question: {exercise}.
Instruction: {instruct}
Hint: {hint}
```

```
Required response format: use a JSON format with the following structure: {{"response":
"Write your answer here..."}}
Please provide a clean and readable output. I insist on this. Do not make formatting
errors.
Respect the output format.'''}
```

A.3.3. Prompt for classifying hints

```
{"role":"system", "content":"You are an expert in teaching mathematics"},
{"role": "user", "content":'''Your task is to verify if a student's revised answer
to a mathematics exercise is correct or not by comparing it with the correct
answer(s) provided. The exercises may have either a single correct answer or
multiple correct answers.

The correct answer(s) for the exercise: {answer}
The student's revised answer: {revised_response}
The hint: {hint}
 1)- If the student's revised answer does not match the correct answer or any
of the correct answers (if multiple), then put the hint in the "wrong_hint" field
of the output.
 2)- If the student's revised answer includes at least one correct answer or
all the correct answers, put the hint in the "correct_hint" field of the output.
I insist on this, please follow this criterion.
 3)- If the hint contains the correct answer(s) or parts of the correct answer(s),
then put the hint in the "wrong_hint" field of the output.
Put the output in a JSON format with the following structure: {{"correct_hint":"","wrong_hi
Make sure that the generated output does not contain escape characters such as
line breaks (\\n) or slashes (\\).
Please provide a clean and readable output. I insist on this. Do not make any
formatting errors.
Follow the output format, and also follow the evaluation criteria and your role.
Do not add anything else.'''}
```

A.3.4. Prompt for checking if the answer is correct and detecting the type of error

```
{"role": "system","content": "You are an expert in teaching mathematics"},
{"role": "user","content": '''Your task is to verify whether a student's answer
to a mathematics exercise is correct or not by comparing it with the correct
answer(s) provided. Exercises may have either a single correct answer or multiple
correct answers.
The correct answer(s) for the exercise: {answer}
The student's answer: {student_answer}
The student's reasoning: {reasoning}
```

```
Categorize the student's error. Here are some categories of errors and examples.
You can add other categories of errors. If the reasoning contains multiple errors,
it is important to list all the present errors.
Specify each error distinctly, even if they belong to different categories or
combine together.
1) Comprehension error: The student does not clearly understand the problem or
the given instructions.
Example: Misreading a problem and confusing the given data.
2) Partial answer: The student provides part of the expected answer but fails
to complete it correctly.
Example: In an equation with two variables, the student finds the value of one
variable but forgets to find the value of the other.
3) Term grouping error: The student incorrectly combines or groups terms in a
mathematical expression.
Example: When simplifying the expression 3x + 2x + 5, the student combines the
terms 3x and 2x to get 5x^2 instead of 5x.
4) Simplification error: The student incorrectly simplifies a mathematical expression.
Example: When simplifying 6x/2, the student divides both the numerator and denominator
by x instead of 2, resulting in an incorrect simplification of 6/2x.
5) Calculation error: The student incorrectly performs mathematical operations.
Example: When multiplying 7 by 8, the student gets 54 instead of 56.
6) Incorrect substitution error: The student substitutes an incorrect value into
an expression or equation.
Example: In the equation 2x + 3y = 10, the student substitutes x = 4 instead
of y = 2, leading to an incorrect solution.
7) Interpretation error: The student incorrectly interprets the problem's instructions
or data.
Example: In a probability problem, the student confuses the probability of event
A with that of the complementary event of A.
8) Algebraic error: The student makes a mistake in algebraic manipulations, such
as distributing, factoring, or solving equations.
Example: In solving 2(x + 3) = 10, the student incorrectly divides 10 by x +
3 instead of 2, leading to an incorrect answer.
1) - If the student's answer does not match the correct answer or any of the
correct answers (if multiple), categorize the type of error and leave the "correct_answer"
field empty.
2) - If the student's answer includes at least one correct answer or all the
correct answers, put the student's answer in the "correct_answer" field. I insist
on this, please follow this criterion.
Put the output in a JSON format with the following structure: {{"error_type":
"","correct_answer": ""}}
Follow the output format, and also follow the evaluation criteria and your role.
Do not add anything else.
    '''}
```

A.3.5. Prompt for obtaining the diverse dataset with different reasoning and answers per type of errors

```
{"role": "system", "content": "You are an expert in teaching mathematics"},
{"role": "user", "content": f'''
Your task is to classify a list of reasonings that contain multiple categories
of errors. For each error category, you must provide the best examples with different
reasoning.
In each reasoning, there may be multiple error categories. If that's the case,
then you must find examples that are different for that group of error categories.

The error categories are already provided in the reasonings.
For each error category or group of error categories, you need to identify and
provide the best k examples of different reasoning.
### Output Format
Make sure the generated output does not contain escape characters such as line
breaks (\\n) or slashes (\\).
Please provide a clean and readable output. I insist on this. Do not make any
formatting errors. Do not add errors that are not in the list.
You must provide the output in JSON format with the following structure:

        {{
            "different_reasoning": {{
                "category_1": [
                    {{"gpt_initial_reasoning": "","initial_response": "","evaluation":
""}},
                    ...
                ],
                "category_2": [
                    {{"gpt_initial_reasoning": "","initial_response": "","evaluation":
""}},
                    ...
                ],
                ...
            }}
        }}
The list of reasonings is: {list_reasoning}
Do not repeat the error groups, for example: calculation error, interpretation
error is the same as interpretation error, calculation error.
The final JSON format must accurately reflect the classification you have made.
Please insert each reasoning into the appropriate category without modifying
the content of the reasoning, the initial response, and the evaluation.'''}
```

A.4. Prompt for hint generation

For the generation of hints, there are only a few differences in the user's role in each prompt. The rest of the content is identical, which is why we will include a complete example of one prompt. For the other prompts, we will only provide the user's role definition, specifying that the rest of the prompt follows the same structure.

A.4.1. Baseline prompt

BaselineOne prompt

```
{"role": "system", "content": "You are an expert in teaching mathematics, helping
students solve a math exercise by providing guiding hints following a specific
cognitive approach."},
{"role": "user", "content": '''Your goal is to generate progressive hints to
help students solve an exercise while following the specified cognitive approach.
The hints should be given in increasing order of difficulty and should not reveal
the final solution. The hints should encourage students to think independently
while providing useful guidance. The hints must be in the form of questions,
and they must not reveal the correct answer or any part of it|I insist on this.

The exercise and question: {exercise}.
Instruction: {instruct}
The correct answer to the exercise: {answer}
Guide according to the cognitive approach: {demarche_cog}
Required response format: use a JSON format with the following structure: {{"hints":
["hint1, hint2..."]}}. Do not number the hints.
I insist that you respect the response format and also ensure that the hints
are in the form of questions and follow the specified cognitive approach. Provide
only the hints, do not include any explanations.'''}
```

BaselineTwo prompt

```
"role": "user", "content": '''Your goal is to identify the common mistakes that
students might make and to generate hints in the form of questions that can help
them correct their mistakes and progress in solving the exercise. The hints must
be in the form of questions, and they must not reveal the correct answer or any
part of it, I insist on this...
```

A.4.2. Prompt based on error type

Prompt based on the student's reasoning

```
{"role":"system", "content": "You are an expert in teaching mathematics"},
{"role": "user", "content":'''Your goal is to provide a clear and relevant hint
to the student to help them correct their reasoning mistakes in math exercises.
If the student has the correct answer, propose a hint to reinforce their understanding.
This hint must be in the form of a question. Additionally, the hint must not
include the correct answer to the exercise or any part of it.
```

```
The exercise and question: {exercise}.
Instruction: {instruct}
The correct answer to the exercise: {answer}
The student's reasoning: {gpt_reasoning}
The student's answer: {gpt_response}
Guide according to the cognitive approach: {demarche_cog}
JSON output format:  {{"hint": "Place the hint here without numbering it..."}}
Ensure the generated output does not contain escape characters such as line breaks
(\\n) or slashes (\\).
Please provide a clean and readable output. I insist on this. Do not make any
formatting errors.
Follow the output format. I emphasize that the hint should not be numbered and
must be in the form of a question. '''}
```

Prompt based on the method used by the student

```
"role": "user", "content":'''Your goal is to provide a hint that helps the student
review the method they are using to solve the math exercise. If the student has
a correct method, propose a hint to reinforce their understanding of that method.
This hint must be in the form of a question. Additionally, the hint must not
include the correct answer to the exercise or any part of it...
```

Prompt based on the application of concepts

```
"role": "user", "content":'''Your goal is to provide a hint that helps the student
review the application of mathematical concepts to solve the exercise and find
the correct answer. If the student is applying the concepts correctly, propose
a hint to reinforce their understanding. This hint must be in the form of a question.
Additionally, the hint must not include the correct answer to the exercise or
any part of it...
```

Prompt based on calculations

```
"role": "user", "content":'''Your goal is to provide a hint that helps the student
review the calculations performed to solve the math exercise. If the student's
calculations are correct, propose a hint to reinforce their understanding of
the calculation steps. This hint must be in the form of a question. Additionally,
the hint must not include the correct answer to the exercise or any part of it...
```

Prompt based on problem interpretation

```
"role": "user", "content":'''Your goal is to provide a hint that helps the student
review their interpretation of the math problem. If the student interprets the
problem correctly, propose a hint to reinforce their understanding. This hint
must be in the form of a question. Additionally, the hint must not include the
correct answer to the exercise or any part of it...
```

Prompt combining all the above aspects

```
"role": "user", "content":'''Your goal is to provide a clear and relevant hint
to the student to help them correct their mistakes and improve their answers
in math exercises. This hint must be in the form of a question. Additionally,
the hint must not include the correct answer to the exercise or any part of it.
Consider the following aspects when generating the hint:
   - Reasoning
   - Method
   - Application of concepts
   - Calculations
   - Interpretation of the problem ...
```

Prompt for completing partial answers

```
"role": "user", "content":'''Your goal is to provide a clear and relevant hint
to the student to help them complete their partial answer in math exercises.
This hint must be in the form of a question. Additionally, the hint must not
include the correct answer to the exercise or any part of it...
```

A.5. Metrics and Additional Results

Metrics such as accuracy, error rate, revision success rate, and mean revision error were used to quantify the performance of the models in order to determine the best prompts.

- Accuracy: For each temperature, we calculate the accuracy as the number of correct responses out of the 40 repetitions divided by the number of responses (number of correct+number of incorrect responses).

$$\text{Accuracy} = \frac{\text{Number of Correct Responses}}{\text{number of responses}}$$

- Revision Success Rate: For a specific type of error at a specific temperature, this is calculated as the number of correct responses out of the 10 repetitions divided by the number of responses.

$$\text{Revision Success Rate} = \frac{\text{Number of Correct Responses}}{\text{number of responses}}$$

- Error Revsion Rate: The revision error rate is calculated as:

$$\text{Error Revision Rate} = 1 - \text{Revision Success Rate}$$

- Mean Revision Error Rate: For all encountered errors at a specific temperature, we calculate the mean revision error rate as the sum of individual error rates divided by the number of errors encountered.

$$\text{Mean Revision Error Rate} = \frac{\sum_{i=1}^{N} \text{ErrorRevisionRate}_i}{N}$$

where N is the number of different types of errors encountered.

A.5.1. Additional results across exercises based on temperature for student model accuracy before and after hints guided by Llama-3-8B-Instruct vs. GPT-4o

Comparison of accuracy before and after providing hints across each exercise for different student models using different teachers models with the best specialized and best baseline-type hint generation prompts is shown in Figure 5 and 4.

A.5.2. Additional results on mean revision error rate by temperature and prompt for each student model across other exercises

The comparison is shown in Figure 8, 7 and 6.

A.5.3. Additional results on the distribution of error revision rates by prompt and temperature for each student model across other exercises

The comparison is shown in Figure 10, 11 and 9.

A.6. Description of exercises and pedagogical elements

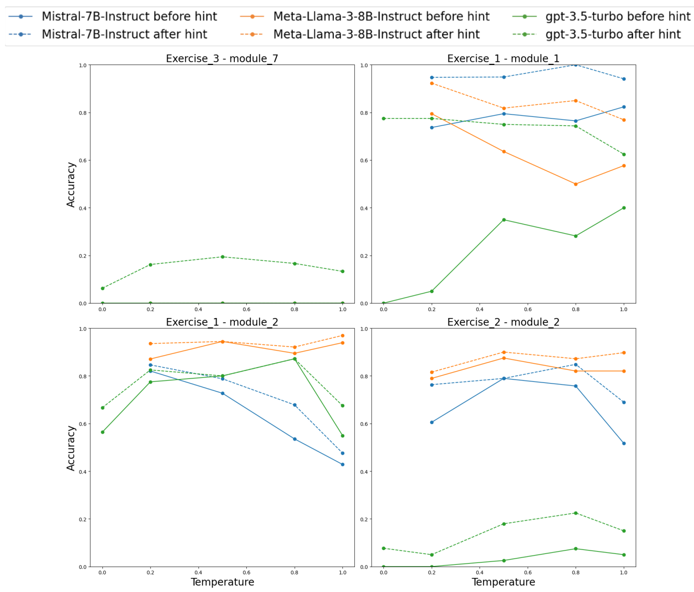
This section presents the exercises used in our study, including key pedagogical elements such as the cognitive approach associated with each exercise, the type of exercise, the exercise statement, instructions, and the corresponding answer. The exercises were originally written in French, and the experiments were conducted in French as well. The prompts were also written in French. For the purpose of this paper, we have translated them into English.

A.6.1. Exercise 1 - module 1

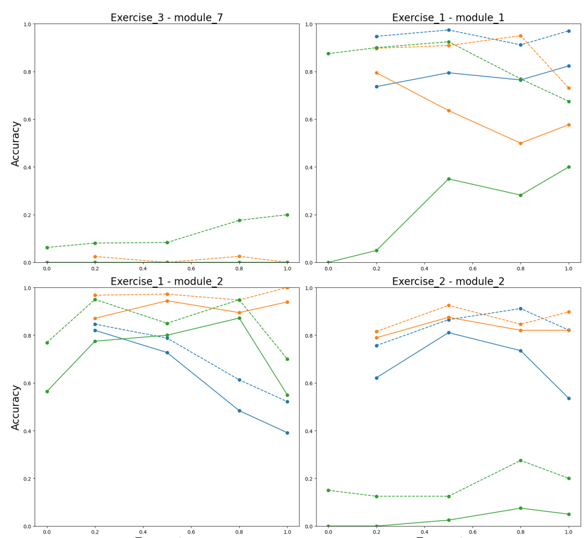
Cognitive Approach: Transition from the concept of partitioning to the concept of fraction as a quotient, through the imposition of a constraint on the whole.
Level 1
This series of activities (A7, A8, A9) aims to gradually move beyond the intuitive notion that an equitable division of a whole composed of multiple units requires taking an equal part of each unit. Starting from Level 2, a condition imposed in the problem statement "forces" the student to counter this conception. The goal is to progressively reach the understanding of a fraction as a quotient. At Level 1, the statement aligns with the student's intuitive conception, with no imposed conditions.

Type of Exercise: The student is presented with a problem involving the division of a whole, composed of n units, into m parts.
Example: The whole consists of 4 units, represented by 4 wooden planks. The whole is divided into 3 equal parts, and the student is asked to interpret the value of one part's size, given the condition of equal portions of each unit. The statement allows the student with a partition-based understanding of fractions to solve the problem by reasoning as follows: $\frac{4}{3}$ is like 4 times one-third of 1.

Exercise Statement:
"Elias bought two quiches of the same size. He decides to eat one-quarter of the quiches

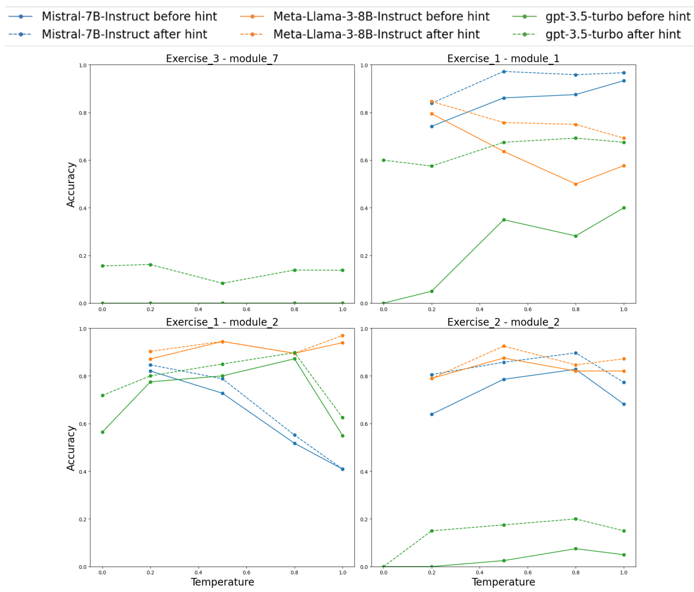


(c) GPT-4o

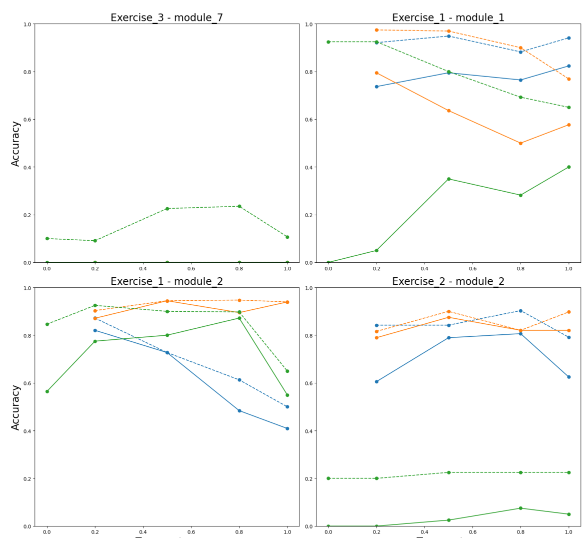


(d) Llama-3-8B-instruct

Figure 4: Comparison of accuracy before and after providing hints across each exercise for different student models, using GPT-4o and Llama-3-8B-instruct as teacher models with **the best specialized hint generation prompt focused on calculation errors**. The results show improved performance when using Llama-3-8B-instruct as the teacher model, but the student models struggled to correct themselves on exercise 3 - module 7.

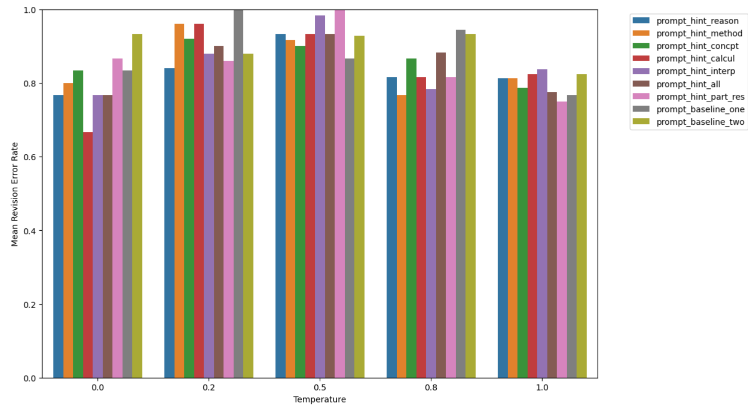


(*a*) GPT-4o

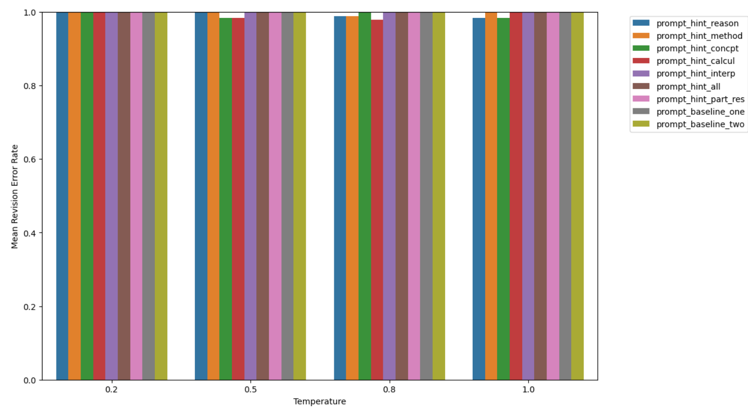


(*b*) Llama-3-8B-instruct

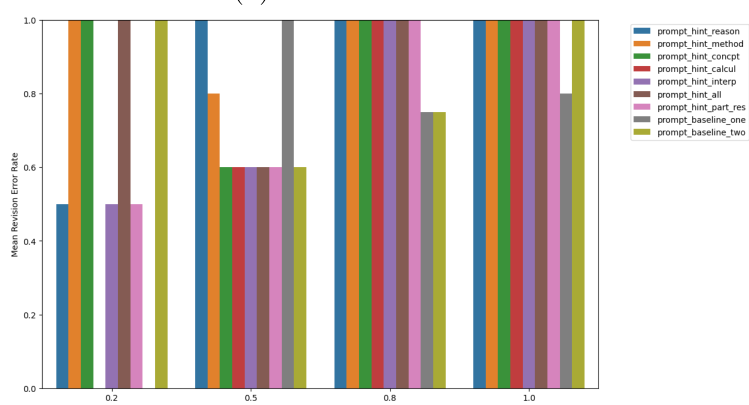
Figure 5: Comparison of accuracy before and after providing hints across each exercise for different student models, using GPT-4o and Llama-3-8B-instruct as teacher models with **the best baseline-type hint generation prompt, named BaselineTwo.** The results show improved performance when using Llama-3-8B-instruct as the teacher model, but the student models struggled to correct themselves on exercise 3 - module 7.



(a) GPT-3.5-turbo

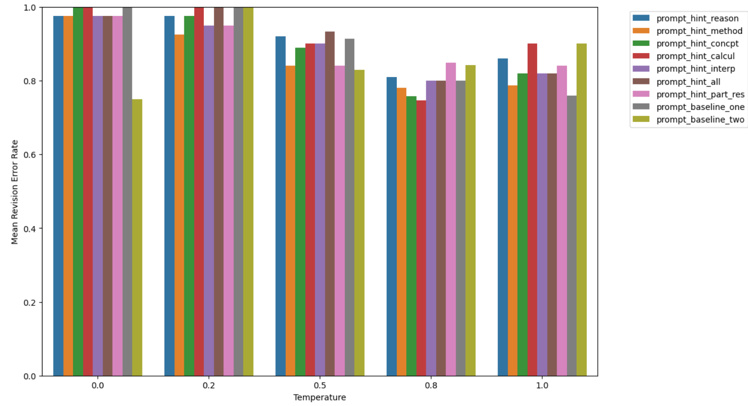


(b) Llama-3-8B-instruct

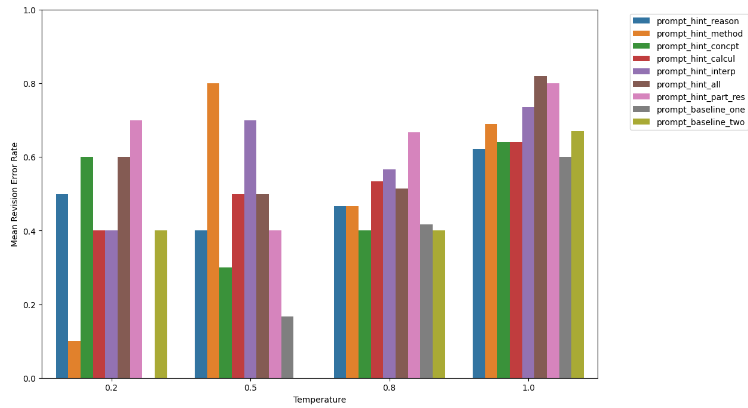


(c) Mistral-7B-Instruct-v0.3

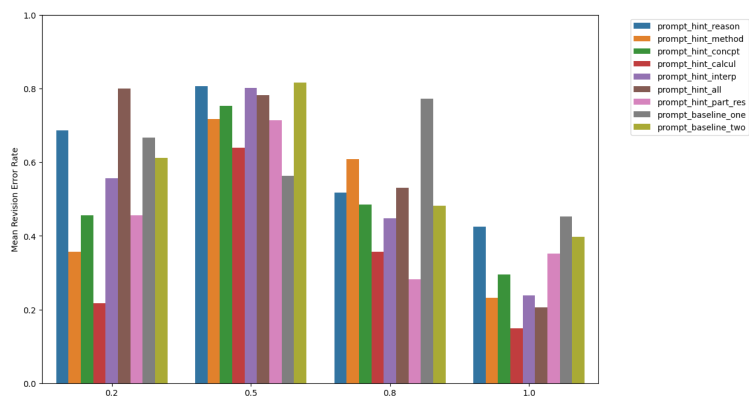
Figure 6: Comparison of mean revision error rates across different temperatures and prompts for hint generation in three student models (GPT-3.5-turbo, Llama-3-8B-instruct, Mistral-7B-instruct-v3) on exercise 3 - module 7. This Figures hows that this particular exercise is challenging to solve, even with hints, for all student models. The accuracies before revision are 0.00 for all temperatures and models: **GPT-3.5-turbo**, **Llama-3-8B-instruct**, and **Mistral-7B-instruct-v3**



(a) GPT-3.5-turbo

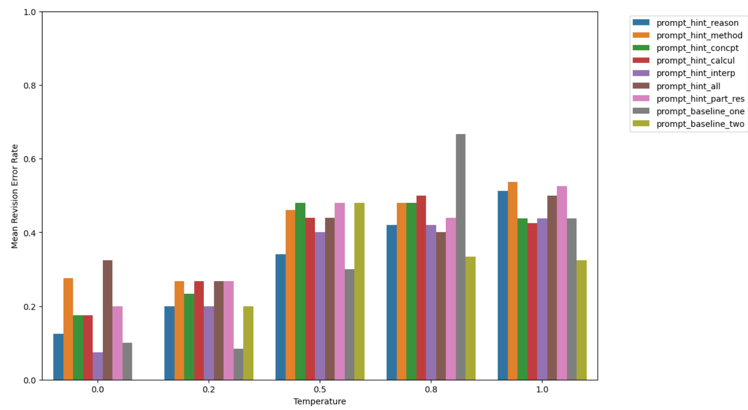


(b) Llama-3-8B-instruct

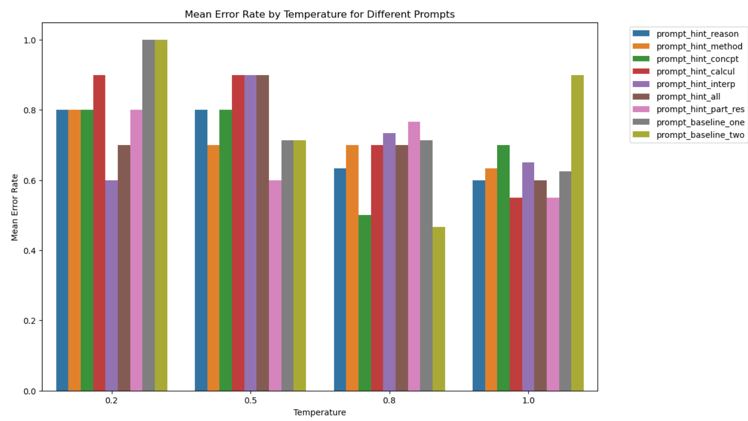


(c) Mistral-7B-Instruct-v0.3

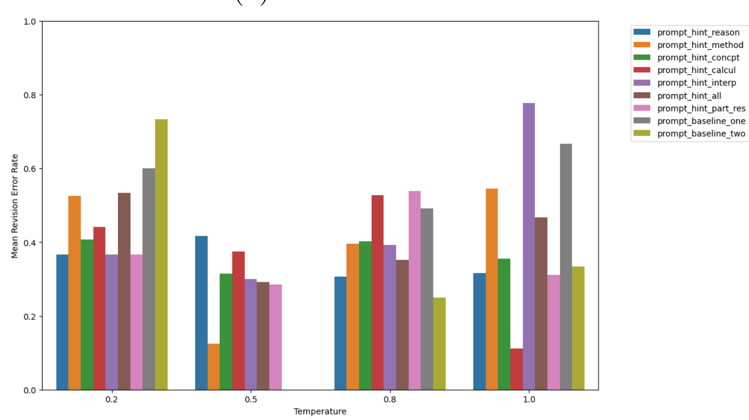
Figure 7: Comparison of mean revision error rates across different temperatures and prompts for hint generation in three student models (GPT-3.5-turbo, Llama-3-8B-instruct, Mistral-7B-instruct-v3) on exercise 2 - module 2. The Figure illustrates how each model's mean revision error rate evolves as the temperature increases and with varying prompts, highlighting how effectively the models can correct themselves using the provided hints. The accuracies before revision are as follows: For **GPT-3.5-turbo**, the accuracies are 0.00 at a temperature of 0.0, 0.00 at 0.2, 0.025 at 0.5, 0.075 at 0.8, and 0.05 at 1.0. For **Llama-3-8B-instruct**, the accuracies are 0.816 at 0.2, 0.875 at 0.5, 0.821 at 0.8, and 0.9 at 1.0. **Mistral-7B-instruct-v3** exhibits accuracies of 0.605 at 0.2, 0.789 at 0.5, 0.714 at 0.8, and 0.517 at 1.0.



(a) GPT-3.5-turbo

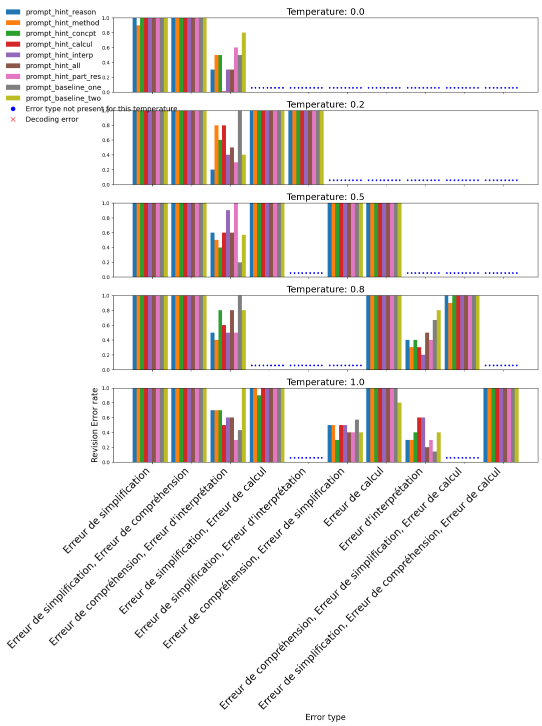


(b) Llama-3-8B-instruct

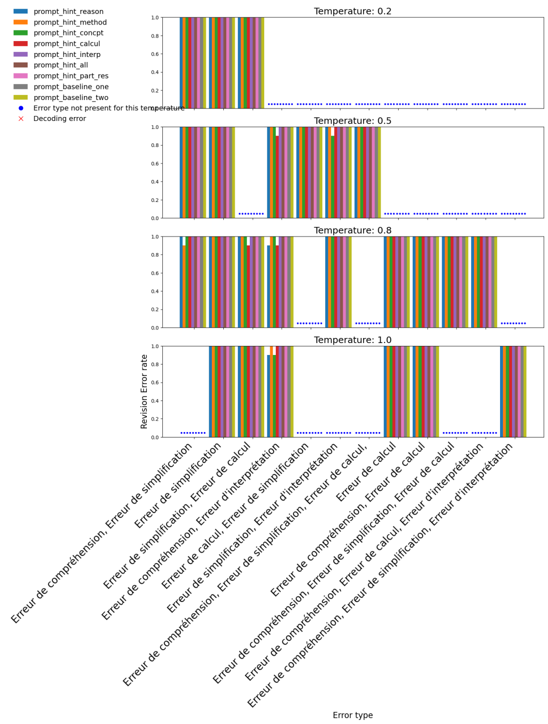


(c) Mistral-7B-Instruct-v0.3

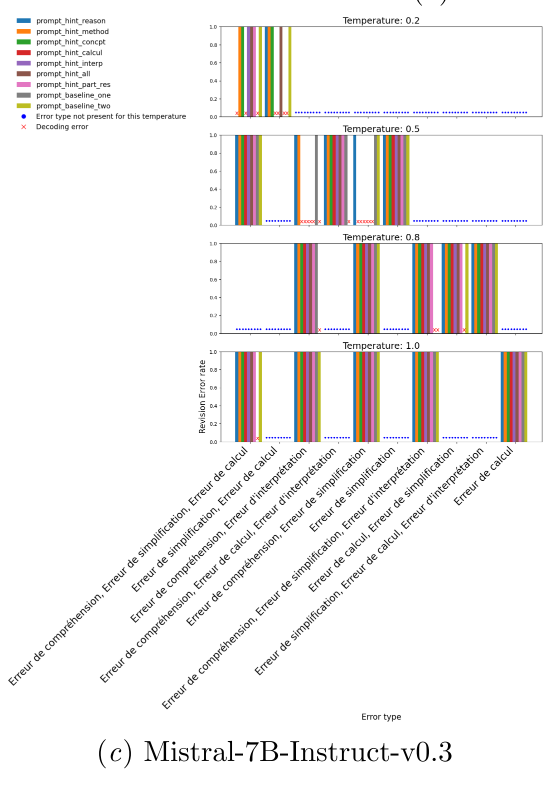
Figure 8: Comparison of mean revision error rates across different temperatures and prompts for hint generation in three student models (GPT-3.5-turbo, Llama-3-8B-instruct, Mistral-7B-instruct-v3) **on exercise 1 - module 1**. The Figure illustrates how each model's mean revision error rate evolves as the temperature increases and with varying prompts, highlighting how effectively the models can correct themselves using the provided hints. The accuracies before revision are as follows: For **GPT-3.5-turbo**, the accuracies are 0.00 at a temperature of 0.0, 0.05 at 0.2, 0.35 at 0.5, 0.282 at 0.8, and 0.4 at 1.0. For **Llama-3-8B-instruct**, the accuracies are 0.795 at 0.2, 0.636 at 0.5, 0.5 at 0.8, and 0.577 at 1.0. **Mistral-7B-instruct-v3** exhibits accuracies of 0.737 at 0.2, 0.795 at 0.5, 0.765 at 0.8, and 0.824 at 1.0.



(a) GPT-3.5-turbo

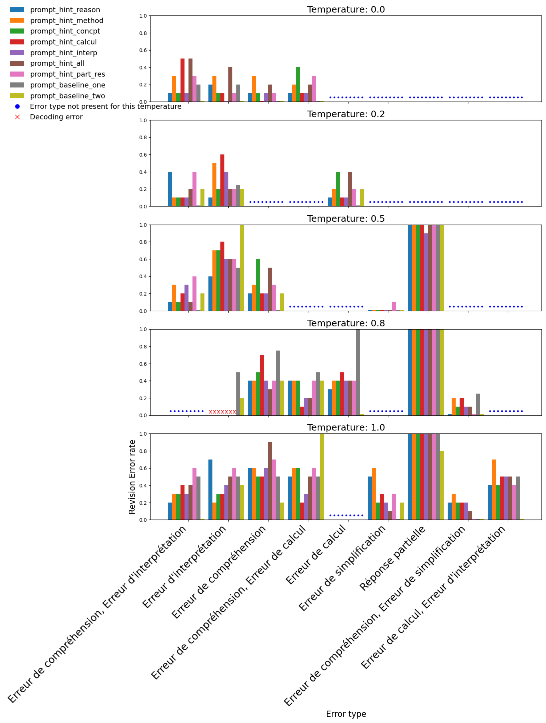


(b) Llama-3-8B-instruct

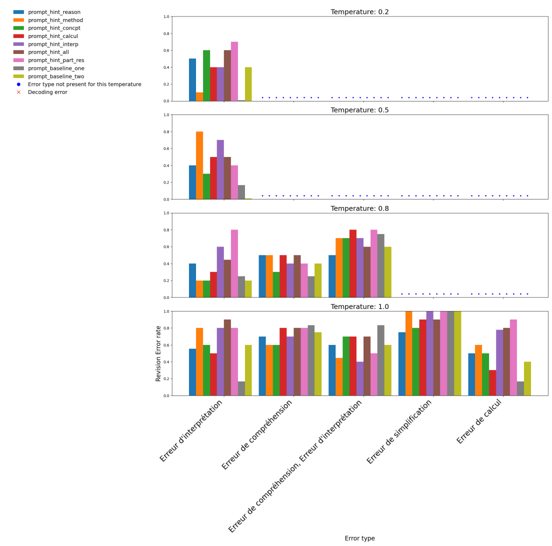


(c) Mistral-7B-Instruct-v0.3

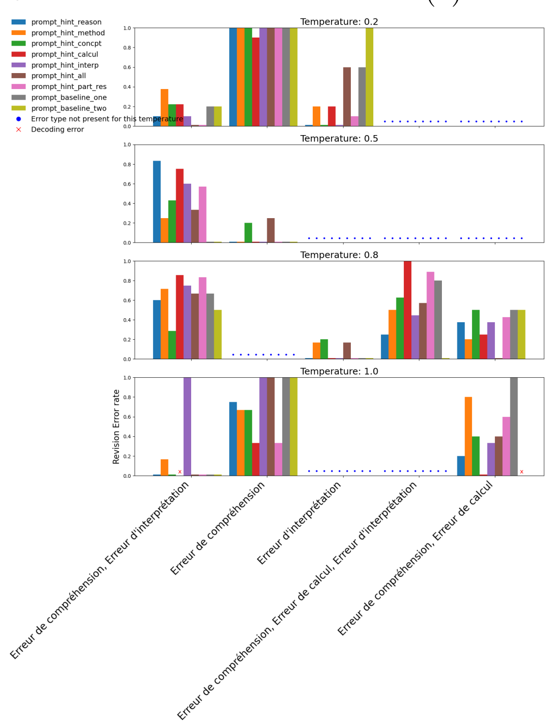
Figure 9: Comparison of error revision rates across different temperatures and prompts for hint generation in three student models (GPT-3.5-turbo, Llama-3-8B-instruct, Mistral-7B-instruct-v3) on exercise 3 - module 7 with GPT-4o as teacher. The figure demonstrates that this exercise is challenging to solve, as indicated by the high error revision rate across all temperatures for all student models except GPT-3.5-Turbo, which attempts to self-correct using the hints.



(a) GPT-3.5-turbo

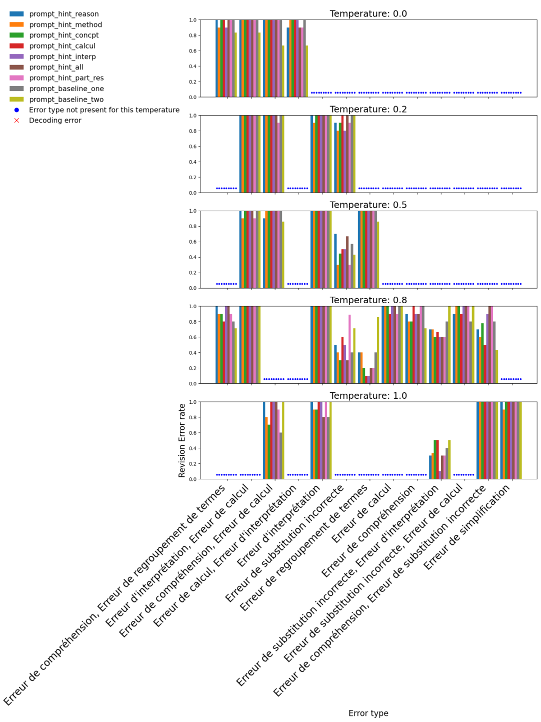


(b) Llama-3-8B-instruct

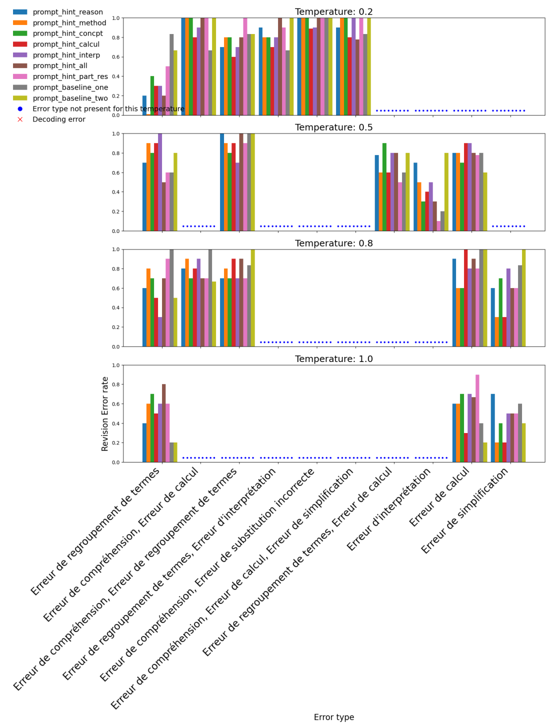


(c) Mistral-7B-Instruct-v0.3

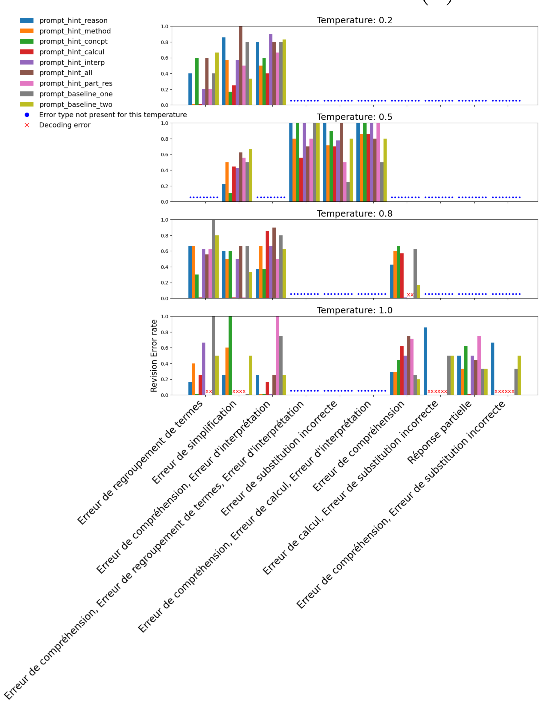
Figure 10: Comparison of error revision rates across different temperatures and prompts for hint generation in three student models (GPT-3.5-turbo, Llama-3-8B-instruct, Mistral-7B-instruct-v3) on exercise 1-module 1. The Figure illustrates how each model's error revision rate changes with increasing temperatures and with respect to types of errors, showing that the frequency of error types tends to increase at higher temperatures.



(*a*) GPT-3.5-turbo



(*b*) Llama-3-8B-instruct



(*c*) Mistral-7B-Instruct-v0.3

Figure 11: Comparison of error revision rates across different temperatures and prompts for hint generation in three student models (GPT-3.5-turbo, Llama-3-8B-instruct, Mistral-7B-instruct-v3) on exercise 2 - module 2 with GPT-4o as teacher. The Figure illustrates how each model's error revision rate changes with increasing temperatures and with respect to types of errors, showing that the frequency of error types tends to increase at higher temperatures.

and save the rest for later. He wants to eat an equal part of each quiche. **What fraction of each quiche will he eat?**"

Instruction:

Complete the following sentence with fractions: Elias will eat ____ of the first quiche and ____ of the second quiche.

Answer:

Elias will eat $\frac{1}{4}$ of the first quiche and $\frac{1}{4}$ of the second quiche.

A.6.2. Exercise 2 - module 2

Cognitive Approach: Adopt a dual perspective to model a multi-step algebraic problem using a literal expression represented both as a sum and as a product.

Level 4

This series of activities (A1, A2, A3, A4) encourages flexibility in problem-solving strategies, moving beyond the strategy suggested by the problem's context and enabling the student to consider an alternative strategy based on the distributive property. This activity reinforces mastery of this property. Depending on the problem scenario, the student's intuitive approach might involve modeling with a literal expression in either expanded form (sum of expressions) or factored form (product of expressions).

At Level 4, one step in the problem requires expressing one variable in terms of another, with an additional challenge introduced as the relationship between these two variables is expressed as a ratio. This ratio involves either multiplying by a fraction less than 1 or dividing by a whole number. For example, $\frac{1}{6}$ of a tulip corresponds to 1 rose.

Type of Exercise: The student is asked to model a two-step problem using a literal expression by selecting one or more correct answers from four given options.

Exercise Statement:

"To decorate her house, Julie enters a store and buys 5 of each of the following items: green plants and flower pots. The price of a green plant varies depending on the store's stock. A green plant costs 3 times as much as a matching flower pot.

Let p be the price of a green plant. **How much did Julie pay in total?**"

Instruction:

Identify the expressions that represent the total price Julie paid.

Select the correct answer(s): $5p + \frac{5p}{3}$? $5(p + \frac{p}{3})$? $5p + \frac{p}{3}$?

Answer:

$5p + \frac{5p}{3}$ and $5(p + \frac{p}{3})$.

A.6.3. Exercise 1 - module 2 (Similar to the previous module, only the exercise is presented here)

Exercise Statement:

"In a restaurant, there are 30 tables. Each table has a bouquet. The number of roses in each bouquet varies depending on the arrivals. Each bouquet contains three times fewer roses than tulips.

Let r be the number of roses. **How many flowers are there in total in this restaurant?**"

Instruction:
Identify the expressions that represent the total number of flowers in the restaurant.
Select the correct answer(s): $30(3r + r)$? $120r$? $30(\frac{r}{3} + r)$?

Answer:
$30(3r + r)$ and $120r$.

A.6.4. Exercise 3 - module 7

Cognitive Approach: Understand how to simplify a fraction to its irreducible form.
Level 1
This series of activities (A3, A4) focuses on using prime factorization to determine whether a fraction is in its simplest form. It aims to develop conceptual expertise in fractional operations and explore the different meanings of fractions in problem-solving contexts. At Level 1, the scenarios align with the intuitive understanding of a fraction as a ratio between a part and a whole.

Type of Exercise: The student answers (yes or no) a question regarding the simplifiability of the fraction presented in the problem. If the fraction is reducible, the student writes its simplified form and completes a response sentence.

Exercise Statement:
"A truck driver covered five thousand five hundred thirty-thirds of a kilometer in two hours."

Instruction:
Can the fraction that describes the number of kilometers the truck driver covered be simplified? Yes? No? If yes: If the truck driver covered five thousand five hundred thirty-thirds of a kilometer in two hours, that means he covered _____ kilometers in two hours.

Answer:
The correct answer is yes. If the truck driver covered five thousand five hundred thirty-thirds of a kilometer in two hours, that means he covered $\frac{500}{3}$ kilometers in two hours.

A.7. Example of good and poor revision of student models on exercise 1 - module 1 after receiving a hint from the teacher model using one of the best prompts

The conversation between teacher and student is in French in our experiments, but we have translated it into English for the purposes of this paper.

- **Teacher Model: Llama-3-8B-instruct**
  - **Student Model: GPT-3.5-turbo**

    Initial Reasoning: Elias decided to eat a quarter of both quiches, so he will eat 1/4 of each quiche. Since he wants to eat an equal part of each quiche, he will eat half of the portion he took from each quiche, which is 1/2 * 1/4 = 1/8 of each quiche.

    Initial Answer: Elias will eat 1/8 of the first quiche and 1/8 of the second quiche.

Teacher Model Hint: How can you use the notion of a quarter to break down the portion Elias eats from each quiche?

Revised Answer (True): Elias will eat 1/4 of the first quiche and 1/4 of the second quiche.

Teacher Model Hint: Can you explain why it is reasonable to think that Elias will eat the same portion of each quiche, even if it means he doesn't eat the entire quarter of the quiches?

Revised Answer (False): Elias will eat 1/8 of the first quiche and 1/8 of the second quiche.

- **Student Model: Mistral-7B-v0.3**

  Initial Reasoning: Elias decided to eat a quarter of each quiche. So he ate half of the first quiche and half of the second quiche.

  Initial Answer: Elias will eat 1/2 of the first quiche and 1/2 of the second quiche.

  Teacher Model Hint: What does it mean if Elias decided to eat a quarter of the quiches, not an equal part of each quiche?

  Revised Answer (True): Elias will eat 1/4 of the first quiche and 1/4 of the second quiche.

  Teacher Model Hint: Can you explain why the 1/4 answer is justified by the phrase "He decides to eat a quarter of the quiches"?

  Revised Answer (False): Elias will eat 1/2 of the first quiche and 1/2 of the second quiche.

- **Student Model: Llama-3-8B-instruct**

  Initial Reasoning: Elias bought two quiches of the same size and decided to eat a quarter of the quiches. This means he will eat 1/4 of the total quiches. To eat an equal part of each quiche, he will eat 1/4 / 2 = 1/8 of each quiche.

  Initial Answer: "First quiche": "1/8"
  "Second quiche": "1/8"

  Teacher Model Hint: Can you explain why you divided a quarter by two to find the portion of each quiche that you will eat?

  Revised Answer (True): Elias will eat 1/4 of the first quiche and 1/4 of the second quiche.

  Teacher Model Hint: How can we think of sharing the quiches as an equitable distribution of the entire quiches, even if the portions are not equal?

  Revised Answer (False): Elias will eat 1/2 of the first quiche and 1/2 of the second quiche.

- **Teacher Model: GPT-4o**

  - **Student Model: GPT-3.5-turbo**

    Initial Reasoning: Elias decided to eat a quarter of both quiches, so he will eat 1/4 of each quiche. Since he wants to eat an equal part of each quiche, he will eat half of the portion he took from each quiche, which is 1/2 * 1/4 = 1/8 of each quiche.

Initial Answer: Elias will eat 1/8 of the first quiche and 1/8 of the second quiche.

Teacher Model Hint: How can you divide what he will eat into equal parts coming from two identical quiches?

Revised Answer (True): Elias will eat 1/4 of the first quiche and 1/4 of the second quiche.

Teacher Model Hint: If Elias eats a quarter of the quiches, what fraction of each quiche does that represent?

Revised Answer (False): Elias will eat 1/8 of the first quiche and 1/8 of the second quiche.

– **Student Model: Llama-3-8B-instruct**

Initial Reasoning: Elias bought two quiches of the same size and decided to eat a quarter of the quiches. This means he will eat 1/4 of the total quiches. To eat an equal part of each quiche, he will eat 1/4 / 2 = 1/8 of each quiche.

Initial Answer: "First quiche": "1/8"
"Second quiche": "1/8"

Teacher Model Hint: What fraction of each quiche should he eat if he wants the sum of the fractions eaten from both quiches to equal a quarter of the total quiches?

Revised Answer (True): Elias will eat 1/4 of the first quiche and 1/4 of the second quiche.

Teacher Model Hint: Elias eats an equal part of each quiche and consumes a quarter in total. How can you divide the total consumption for each quiche?

Revised Answer (False): Elias will eat 1/8 of the first quiche and 1/8 of the second quiche.

– **Student Model: Mistral-7B-v0.3**

Initial Reasoning: Elias decided to eat a quarter of each quiche. So he ate half of the first quiche and half of the second quiche.

Initial Answer: Elias will eat 1/2 of the first quiche and 1/2 of the second quiche.

Teacher Model Hint: How can Elias divide the total amount he wants to eat, which is a quarter of all the quiches, equally between the two quiches?

Revised Answer (True): Elias will eat 1/4 of the first quiche and 1/4 of the second quiche.

Teacher Model Hint: What would the calculation be if Elias decided to eat a quarter of the total amount of the two quiches together rather than eating a quarter of each quiche individually?

Revised Answer (False): Elias will eat 1/2 of the total amount of the two quiches.