# ProbLLMs: Multiple-Choice Problem Solving for EPFL Courses

Arvind S. Menon | 354584 | arvind.menon@epfl.ch
Lars C.P.M. Quaedvlieg | 352130 | lars.quaedvlieg@epfl.ch
Lazar Milikic | 353622 | lazar.milikic@epfl.ch
Somesh Mehra | 353628 | somesh.mehra@epfl.ch
GradientVanishers

## Abstract

Large Language Models (LLMs) have the potential to revolutionize education by increasing the accessibility of personalized and on-demand learning assistance. However, in the fields of Science, Technology, Engineering, and Mathematics (STEM), which often require complex reasoning, general-purpose LLMs typically underperform. This work aims to develop an AI tutor targeted at STEM education, specifically for multiple-choice question answering related to EPFL courses. Using a small general-purpose LLM as a base, we fine-tune a model with enhanced capabilities for complex reasoning tasks related to STEM education. Additionally, we augment our model using knowledge retrieval to improve performance and demonstrate that quantization can viably improve the accessibility of our model in lower-resource settings while maintaining reasonable performance. Our DPO-aligned model outperforms the base version for STEM question answering, bringing us one step closer to a scalable solution for personalized learning assistance in STEM education.

## 1 Introduction

The remarkable capabilities of large language models (LLMs) to understand and interact with humans through natural language have inevitably led to their widespread adoption across various domains. A particularly impactful application is in the field of education. With limited teaching resources being a common issue globally, an AI tutor enabling independent student interaction and providing direct answers could be invaluable. This would not only reduce educators' workload but also improve the accessibility of personalized assistance and promote educational equality worldwide (Kılınç, 2023).

While LLMs have demonstrated strong capacities for commonsense reasoning and a wide array of question-answering tasks (Naveed et al., 2024), they often fall short in more complex reasoning tasks (Bian et al., 2024). This limitation is particularly problematic in STEM (science, technology, engineering, and mathematics) education, where answering questions often requires understanding theorems and complex reasoning. Therefore, using most existing LLMs directly out of the box might not provide a robust AI tutor for many STEM courses, especially at higher education levels.

In this work, we develop an AI tutor capable of answering multiple-choice questions related to EPFL courses in the fields of mathematics, physics, computer science, and electrical engineering. We leverage various datasets to initially tune an existing LLM for better performance on challenging STEM questions, and subsequently adapt this model to generate answers for multiple-choice questions. Additionally, we experiment with augmenting the model with knowledge retrieval to enhance question-answering accuracy, and with model quantization to drastically reduce memory and compute requirements, enabling deployment in more resource-constrained environments.

Our fine-tuned 3.8B-parameter model is able to achieve a higher overall performance than its base model on multiple complex reasoning datasets. We include a qualitative analysis of our method on different domains and show that using Direct Preference Optimization (Rafailov et al., 2024b) can lead to hindering of in-context learning capabilities, which retrieval augmented generation is highly dependent on. We finally propose ways of avoiding this issue.

## 2 Related Work

**Alignment to Human Preferences.** LLMs excel in generating human-like text, yet aligning them to specific objectives such as safety or human preferences necessitates tailored approaches. Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022) represents a fundamental method, refining models based on human-

rated preference datasets. Building on this, Direct Preference Optimization (DPO) (Rafailov et al., 2024b) streamlines the process by directly optimizing for human preferences, offering a more stable and efficient alternative to RLHF. Our work seeks to apply these methodologies in an educational setting, where the accuracy, relevance, and completeness of reasoning are critical so that LLMs can become a valuable learning tool.

**Task-Specific Adaptation.** While LLMs are inherently task-agnostic, adapting them for specific tasks is essential. Model finetuning, which updates model weights via supervised training tailored to a specific task, often requires extensive datasets and can lead to poor generalization and catastrophic forgetting (McCoy et al., 2019; Luo et al., 2024). As an alternative, few-shot in-context learning leverages a few illustrative examples added directly to the prompt without updating model weights, significantly reducing data and training needs (Radford et al., 2019). Moreover, recent studies show that zero-shot prompting, where tasks are described without example inputs, can still yield robust task-specific performance (Kojima et al., 2022).

**Parameter Efficient Fine-Tuning.** Given the considerable size of many pre-trained language models, fully finetuning all parameters is resource-intensive and often impractical. Numerous strategies have been developed for parameter-efficient finetuning. One notable method is Low-Rank Adaptation (LoRA) (Hu et al., 2021), which freezes the original model weights and introduces low-rank trainable matrices at specific layers. This approach achieves results comparable to full fine-tuning with significantly fewer trainable parameters and reduced memory requirements. Unlike other methods such as those proposed by Houlsby et al. (2019), LoRA does not increase latency during inference.

Our project eventually explores lightweight adaptation methods, such as in-context learning and LoRA, within educational applications, aiming to enhance LLM responsiveness without the typical overhead associated with traditional fine-tuning. This approach makes adjusting LLMs more accessible and reduces the need for extensive computational resources.

**Retrieval Augmented Generation (RAG).** Lewis et al. (2021) introduced the RAG approach to enhance LLMs by dynamically integrating context from a knowledge base directly into each prompt. This is particularly vital in educational settings, where the accuracy of information—such as theorems, formulas, or lecture notes—is critical. RAG allows LLMs to provide precise responses relevant to specific courses reducing the need for task-specific adaptation. Notably, RAG has been successfully applied to improve mathematical problem-solving in middle-school education (Levonian et al., 2023) and to enhance medical training in low-income countries (Al Ghadban et al., 2023). Inspired by these applications, our project extends RAG to university-level STEM courses, aiming to boost accuracy and reduce the necessity for extensive model tuning.

**Quantization.** Since large models require heavy computational resources, numerous methods have been developed to reduce their size. Whilst reducing the precision of model parameters is a simple method to achieve this, integer quantization can significantly improve efficiency over floating point inference (Jacob et al., 2018). Recent methods are able to perform post-training quantization to 8-bits with minimal loss in performance (Xiao et al., 2023a), whilst 4-bit quantization is shown to be almost universally optimal for the accuracy versus model size tradeoff (Dettmers and Zettlemoyer, 2023). This is imperative for increasing the accessibility of large models for lower-resource settings, which is particularly relevant for education to help maintain equality.

## 3 Approach

To develop our AI tutor, we first collect relevant, high-quality data for fine-tuning. Then, using Phi-3-mini (Abdin et al., 2024) as a base, we adapt this model for STEM question answering, specifically for multiple choice questions. Additionally, we augment our model using RAG to improve the accuracy of the question answering with external knowledge retrieval, whilst also experimenting with quantization to reduce resource requirements for our model. A high-level overview of our approach is shown in Figure 1.

### 3.1 Dataset Creation

We leverage various sources to collect datasets for training and evaluating our model.

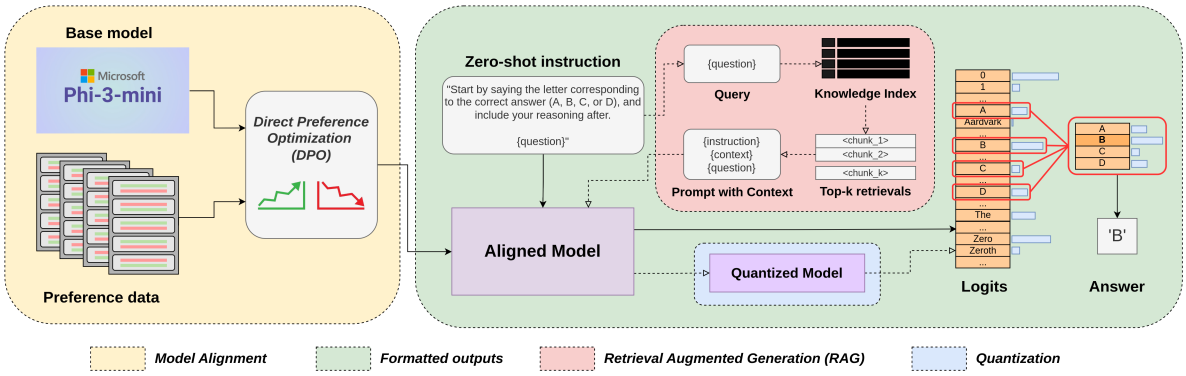


Figure 1: High-level overview of our approach. We start with Phi-3-mini as a base model and apply DPO using our preference datasets to align the model towards better reasoning for STEM question answering (yellow box). To ensure correctly formatted outputs for MCQA, we employ zero-shot prompting, instructing the model to begin its response with the letter corresponding to the answer. Instead of allowing the model to generate full responses and then extracting the letter representing the answer to the MCQA question, we constrain the model to generate only one token. For this token, we take the logits of the four answer options ('A', 'B', 'C', and 'D') and output the letter corresponding to the highest logit. This ensures correctly formatted outputs even when other tokens have higher probabilities (green box). Additionally, we optionally augment the model with RAG. The question is passed as a query to our knowledge index, the top-k results based on a vector search over the embeddings are retrieved, and these results are added as context to the prompt (red box). We also benchmark with a quantized model, which has a significantly reduced memory footprint (blue box).

#### 3.1.1 Preference Data

To align our model for answering EPFL course questions, we have constructed a dataset of preference pairs utilizing multiple sources.

**MNLP Students-Annotated Dataset:** This dataset initially comprised approximately 22,000 preference pair samples, gathered from interactions with ChatGPT. These interactions involved answering questions related to electrical engineering, computer science, machine learning, and physics courses at EPFL. Students annotated the responses based on several criteria including correctness, relevance, clarity, and completeness. For our project, which aims to enhance LLMs' accuracy in responding to EPFL course-related queries, we specifically filtered out any preference pairs where both the chosen and rejected answers were deemed incorrect.

**Computer Science Theory QA Dataset:** This dataset, publicly available on Kaggle (Mateen, 2023), comprises question-answer pairs covering 150 subtopics across diverse Computer Science and Machine Learning disciplines. To construct a preference pair dataset, we adopt the methodology proposed by (Huang et al., 2023), which involves transforming correct answers into rejected ones. Specifically, we employ three ChatGPT-driven prompts to corrupt the answers: *REMOVE* essential content, *SUBSTITUTE* parts of the answer with incorrect information, or *INSERT* irrelevant details. For each transformation, two out of the three corruption techniques are randomly applied to modify the original correct answer into a rejected one. Detailed descriptions of these corruption prompts are available in Appendix A.

**Camel-AI Math & Physics Datasets:** These two datasets contain over 70,000 examples of various math and physics questions (Li et al., 2023). Although the datasets are large, the data was synthetically generated by GPT-4 and may contain inaccuracies as noted by the authors. Each dataset consists of question-answer pairs, allowing us to apply the same corruption techniques described earlier to generate rejected answers. Due to budget constraints with ChatGPT, we sample approximately 2,500 samples from the Camel-AI Math dataset and about 3,000 from the Camel-AI Physics dataset.

**Stack Exchange Preferences Dataset:** This dataset comprises preference data from over 10 million questions sourced from various Stack Exchange forums, including Stack Overflow, Mathematics Stack Exchange, Physics Stack Exchange, Computer Science Stack Exchange, etc. (Lambert et al., 2023). Each question is associated with multiple answers, receiving a preference score based

on user upvotes. We selectively compile preference pairs by selecting a "chosen answer" that not only has the highest preference score among the answers but also possesses a minimum score of 10. Simultaneously, we identify a "rejected answer" with a score between 3 and 5. This selection criterion ensures a high-quality "chosen answer" while the "rejected answer" is considered adequate, yet maintaining a significant quality gap between them. We exclude examples that do not meet these conditions. Through this method, we have collected approximately 42,000 samples across various relevant domains.

#### 3.1.2 Evaluation Data

To evaluate the accuracy of our model in generating correct answers, we have gathered various multiple-choice question answering (MCQA) datasets. We combine all available data splits to maximize the number of evaluation examples, as MCQA datasets are not required for training in our approach (Section 3.4).

**ARC:** This dataset contains grade-school level multiple-choice science questions. It is divided into an Easy set and a Challenge set, with 5,196 and 2,590 questions respectively (Clark et al., 2018).

**MMLU:** This dataset includes multiple-choice questions from various topics (Hendrycks et al., 2021a). We retained only a subset of relevant topics in the domains of mathematics, physics, computer science, and engineering, resulting in 2,719 questions. A complete breakdown of topics is provided in Appendix B.

**MATH:** This dataset contains 12,500 challenging competition mathematics problems, each with step-by-step solutions that contain a free-form answer inside a '\boxed{}' element (Hendrycks et al., 2021b). We convert this into an MCQA dataset by synthetically generating three additional answer options and shuffling them. Since the answers are not always a single number (for example they can contain fractions, square roots, etc.), to generate reasonable options, we first parse all the numbers from the answer and randomly select one number to add a value between -5 to 5.

**TheoremQA:** This dataset contains challenging university-level questions paired with STEM theorems (Chen et al., 2023), designed to benchmark LLMs' ability to apply theorems to solve questions requiring complex reasoning. To test the LLMs' knowledge and fairly evaluate the impact of model augmentations, we do not provide the accompanying theorems. Instead, we only provide the questions, requiring the model to use its internal knowledge and any context from Retrieval-Augmented Generation (RAG) if provided. Furthermore, we retain only questions that have numeric answers and are from relevant domains (mathematics, electrical engineering, computer science, and physics). The answers can be either a single float or integer, or a list of numbers. If the answer is a single float $x$, we generate three incorrect options as random numbers in the interval $[0.9x,\ 1.1x]$. For integers, the generation of options follows the same method as for the MATH dataset. When the answer is a list of numbers, for each synthetic option, we use the corresponding method for integers or floats to alter a number at a randomly selected list index. Using this approach, we obtain 579 questions.

#### 3.1.3 RAG Data Collection

To construct the knowledge base for our Retrieval Augmented Generation (RAG)-enhanced model, it is crucial to ensure that the content is representative and specifically tailored to accurately answer multiple-choice questions (MCQs) from EPFL courses across several disciplines, including Computer Science and Systems, Artificial Intelligence and Machine Learning, Theoretical Physics, and Electrical Engineering. We primarily utilize the EPFL Moodle and EPFL Study plans platforms as our main resources. Specifically, they jointly provide a categorized list of EPFL courses; we have identified all pertinent courses taught in English within the categories of Electrical and Electronics Engineering (EL), Computer Science (IN), and Physics (PH). A comprehensive list of these courses, selected based on their relevance to the topics addressed by this project and the examples annotated in the initial phases, is detailed in Appendix C.

For each identified course, we collect RAG material primarily from official bibliographies or, if unavailable, from specific lecture notes provided by the instructor. This collection strategy ensures that the materials provided to the RAG are directly relevant to the core content of the selected courses.

### 3.2 Base Model

We utilize Phi-3-mini (Abdin et al., 2024), a model with 3.8 billion parameters, as the base model for the question-answering task. According to key

benchmarks, Phi-3-mini outperforms many larger models in reasoning and logic capabilities. Its strength lies in a specific two-phase training approach. In the first phase, the model learns general knowledge and language understanding primarily from web sources. The second phase incorporates heavily filtered web data (a subset used in Phase-1) along with synthetic data, which teaches the model logical reasoning and various niche skills through a curriculum-style learning process.

### 3.3 Model Alignment

To tailor our general-purpose language model to answer STEM course questions, we align it using our preference datasets described in Section 3.1.1. We employ Direct Preference Optimization (DPO) (Rafailov et al., 2024b), utilizing LoRA (Hu et al., 2021) for parameter-efficient fine-tuning. These methods are implemented using HuggingFace's TRL (von Werra et al., 2020) and PEFT (Mangrulkar et al., 2022).

The DPO objective is defined as follows:

$$\mathcal{L}_{\text{DPO}} = -\log \sigma \left( r_\theta(\mathbf{x}, y^+) - r_\theta(\mathbf{x}, y^-) \right) \ , \quad (1)$$

where $\sigma$ is the sigmoid function, and $r_\theta(\mathbf{x}, \cdot)$ represents the reward model parameterized by $\theta$ applied to the input prompt $\mathbf{x}$. $y^+$ and $y^-$ are responses to the prompt with $y^+$ being preferred over $y^-$.

Furthermore, we define the reward function as

$$r_\theta(x, y) = \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)} + \beta \log Z(x) \ , \quad (2)$$

where $\pi_{\text{ref}}$ is a reference policy, $\pi_\theta$ is the model's policy, $Z(x)$ is the partition function, and $\beta$ is a parameter controlling the deviation from the base reference policy.

As the preference data is annotated such that the preferred answers exhibit better correctness, completeness, and relevancy for questions from relevant STEM courses, aligning the model with DPO is expected to further enhance its reasoning capabilities for STEM question answering.

LoRA is used to fine-tune only a subset of model parameters, significantly reducing the computational cost. The LoRA method modifies the original weights $W$ by introducing low-rank matrices $A \in \mathbb{R}^{m \times r}$ and $B \in \mathbb{R}^{r \times n}$, where $r < m, n$ is the LoRA rank:

$$W' = W + \Delta W = W + AB \ . \quad (3)$$

In Rafailov et al. (2024a), the authors describe that when performing supervised fine-tuning (SFT) on the reference model $\pi_{\text{ref}}$, the implicit rewards of both the chosen and rejected response decline, though the margin between them increases. However, when one does not SFT before DPO, there is little discernible trend in the average implicit reward and the implicit rewards of the chosen responses remain above zero, which might lead to model divergence. However, in our experiments, performing DPO without any SFT on the reference policy $\pi_{\text{ref}}$ did not result in any divergence issues.

### 3.4 Formatted Outputs

To specialize our model for multiple choice question answering (MCQA), it is essential to ensure that the outputs are correctly formatted as a single letter corresponding to the correct answer option. Our model is tuned to provide not only correct answers but also reasoning that is complete, relevant, and clear. Therefore, we need an approach that retains the benefits of instructing the model to reason about the answer (Kojima et al., 2022) while ensuring the output is a single letter.

We propose two approaches to ensure the output is in the proper format: Letter Extraction and MCQ-Logits.

**Letter Extraction:** In this approach, we examine the first 10 tokens generated by the model. If one of these tokens corresponds to one of the four answer options ('A', 'B', 'C', or 'D'), we set that token as the model's output. If none of these tokens match the answer options, we default to 'C'.

**MCQ-Logits:** We restrict the model to generate only one token and then extract the logit values for the corresponding answer options. The model outputs the letter corresponding to the highest logit value. This method ensures correctly formatted outputs, even when other tokens have higher probabilities or when the model does not provide the answer in the form of a letter. For instance, the model might generate a full answer instead of the answer letter, despite being instructed otherwise. The green box in Figure 1 illustrates the MCQ-Logits approach.

For both approaches, we instruct the model to generate a letter corresponding to the correct answer by relying on in-context learning (Brown et al., 2020). Additionally, to harness the power of chain-of-thought prompting (Wei et al., 2023), we devise the following prompt:

*"Start by saying the letter corresponding*

*to the correct answer (A, B, C, or D), and include your reasoning after.*
*Question: [Text of the question]*
*Answer:"*

Since we only need the letter of the answer and not the reasoning, we stop the generation after 10 tokens for Letter Extraction and after one token for MCQ-Logits. This approach makes the model think it should reason when generating, but we do not let it finish, achieving what we call **Truncated Reasoning**, where we ideally obtain the benefits of chain-of-thought reasoning without the overhead of generating full answers.

We also employ few-shot in-context learning to improve the model's performance (Xie et al., 2022). Specifically, we provide several examples demonstrating that the model should generate only the letter corresponding to the selected answer after *"Answer:"*. The few-shot prompt we use is provided in Appendix D.1.2. Additionally, to fully harness the power of chain-of-thought prompting, we include reasoning for each example question after the answer as given in Appendix D.1.3. This ensures that the model first answers with the selected letter and then reasons about it, so we can apply Truncated Reasoning.

### 3.5 Model Augmentations

#### 3.5.1 Retrieval Augmented Generation (RAG)

To enhance the accuracy of our models and fill potential knowledge gaps, we augment the prompts with relevant contextual information retrieved from external sources. This method, known as Retrieval Augmented Generation (RAG), is applied to improve our LLMs.

To integrate RAG into our MCQA pipeline, we first index the collected documents (collected as described in Section C) using a vector store index from the LlamaIndex library (Liu, 2022). The vector store index splits the documents into chunks of a specified size, which are then encoded by the BAAI BGE-large sentence transformer (Xiao et al., 2023b). Once all the documents and their corresponding chunks are encoded and the document index is created, we store the index on disk. This allows us to avoid re-indexing each time we use our RAG-enhanced LLM.

Then, we extend the pipeline so that before answering a question, we encode it using the BAAI BGE-large model and retrieve the top-k chunks from the vector document index with the most similar embeddings to the posed question. These retrieved chunks are then added as additional context to the prompt. The template for our RAG prompt is provided in Appendix D.2.

#### 3.5.2 Quantization

We utilize HuggingFace's standard 4-bit and 8-bit quantization methods proposed by Jacob et al. (2018). These methods convert the 32-bit floating-point weights of a model into 4- and 8-bit integers. For the original model, we use the bfloat16 datatype for training, which is a 16-bit floating-point format that reduces training time while preserving accuracy during optimization.

## 4 Experiments

### 4.1 Evaluation

**Datasets.** We conduct a series of experiments to evaluate the effectiveness of various large-scale language models across multiple MCQA benchmark datasets, including MMLU, MATH, ARC Challenging, and TheoremQA, which are described in detail in Section 3.1.2.

**Baselines.** We benchmark the Phi-3-mini-4k-instruct model and its different configurations aligned with DPO, referred to as ProbLLM-3.8B.

**Training.** We fine-tune Phi-3-mini-4k-instruct with DPO and LoRA on one NVIDIA A100 (80GB) GPU. Detailed training hyperparameters are provided in Appendix E.

Our baseline comparisons involve multiple prompting techniques and retrieval mechanisms, including zero-shot, three-shot, and specialized reasoning prompts, which are described in Appendix D.1. As discussed in Section 3.4, we experiment with standard answer extraction, referred to as "Letter Extraction", where we find the letter of the selected answer in the generated text. We also use the character with the largest logit among all valid multiple-choice answers, referred to as "MCQ-logits" in the results.

Finally, we explore the impact of quantization on model size and efficiency, comparing full precision against 8-bit and 4-bit quantization.

**Evaluation metrics.** The primary evaluation metrics include accuracy and model footprint, which provide insights into the trade-offs between computational efficiency and model performance. We also provide a qualitative evaluation in Section 5.

### 4.2 Results

From Figure 2, we observe that the MCQ-Logits answer extraction strategy consistently outperforms Letter Extraction. Specifically, MCQ-Logits shows a moderate but persistent improvement in performance across different few-shot prompting techniques. Zero-shot prompting combined with MCQ-Logits achieves the highest average accuracy among all datasets, performing slightly worse on general reasoning benchmarks but excelling on math-related datasets. For more details, refer to Appendix Table 5.

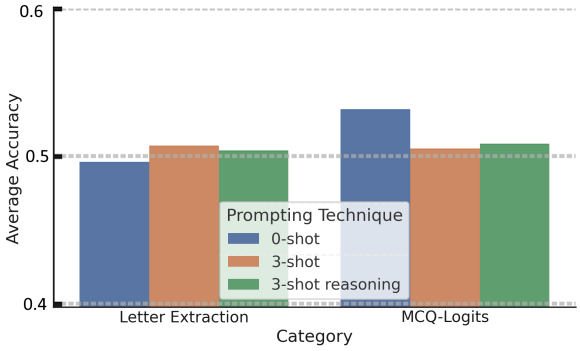


Figure 2: Average accuracy achieved by our models over all datasets per prompting technique and answer extraction strategy.

Interestingly, despite expectations that few-shot examples would help the model adjust its logits better—by showing examples where the first generated token is always a letter corresponding to the answer—the results favor zero-shot prompting. Therefore, we opt for a zero-shot prompting approach, directly extracting the answer from the logits of the answer characters for all further model evaluation and improvement strategies.

We then benchmark our model, ProbLLM-3.8B, against Phi-3-mini-4k-instruct. As shown in Table 1, our DPO-aligned model outperforms the base Phi-3-mini-4k-instruct across all datasets. These results demonstrate that DPO alignment has successfully enhanced the model's knowledge and reasoning capabilities.

#### 4.2.1 Experiments with Retrieval Augmented Generation (RAG)

We experimented with various RAG hyperparameters, particularly focusing on the number of chunks to retrieve from the document index for each question.

As shown in Figure 3, the model's performance improves when using up to three of the most similar chunks. Beyond this point, performance starts to degrade, likely due to the inclusion of less relevant context, which introduces noise and limits the utility of important information. Although the optimal number of chunks may vary for different datasets, we found that using three chunks generally provided the best results. Therefore, we use this setting throughout the remainder of our evaluation.

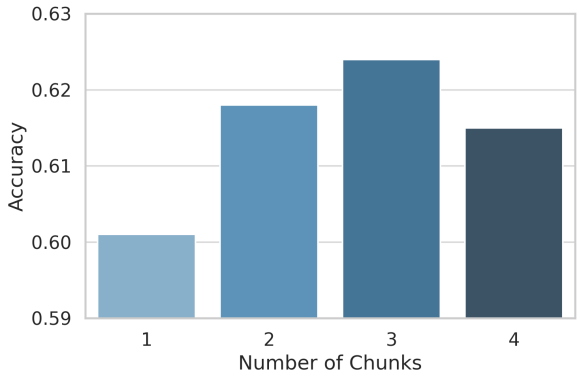


Figure 3: Performance of the model with RAG on the MMLU dataset, using different numbers of the most similar chunks to retrieve the context for augmenting the model's prompt.

Additionally, we conducted experiments with models enhanced by RAG. Table 2 compares the performance of different configurations and augmentations applied to the ProbLLM-3.8B and Phi-3-mini-4k-instruct models. While ProbLLM-3.8B outperforms Phi-3-mini-4k-instruct without RAG, the trend reverses with RAG enhancement, where Phi-3-mini-4k-instruct achieves somewhat better performance than the DPO-aligned ProbLLM-3.8B. Overall, RAG models enhance performance for math-related benchmarks, achieving the strongest results on MATH. Furthermore, Phi-3-mini-4k-instruct enhanced with RAG is the highest-performing configuration on MMLU.

However, for the ARC Challenging and TheoremQA benchmarks, we observe a surprising and significant drop in performance with RAG-enhanced models, resulting in an overall decline compared to non-augmented models. This could be explained by the nature of these benchmarks, as described in Section 3.1.2, and the way we collected the RAG external knowledge base, which relies on EPFL course materials (Section 3.1.3). Our RAG knowledge base may lack some general theorems and knowledge expected by these bench-

Table 1: Benchmark accuracies of evaluated model performances across different evaluation benchmarks.

| Model | MMLU | MATH | ARC Challenging | TheoremQA |
|---|---|---|---|---|
| Phi-3-mini-4k-instruct | **0.525** | 0.420 | 0.869 | 0.381 |
| ProbLLM-3.8B | **0.525** | **0.424** | **0.897** | **0.392** |

marks, causing the RAG retriever to supply less relevant context for these questions, thus confusing the model more than assisting it. Conversely, the MATH and MMLU benchmarks are more closely aligned with the material we expect for STEM EPFL questions, allowing the RAG retriever to provide more relevant context to our LLMs.

#### 4.2.2 Quantization

We experimented with 4-bit and 8-bit quantization to augment the model. Figure 4 illustrates how model performance degrades with different quantization techniques across all datasets. The performance drop is most significant for the math-related datasets, with declines of up to 5%, as shown in Table 6 from Appendix. However, these quantization methods significantly reduce the model's footprint, by factors of 6.8× and 3.7× for 4-bit and 8-bit quantization, respectively.

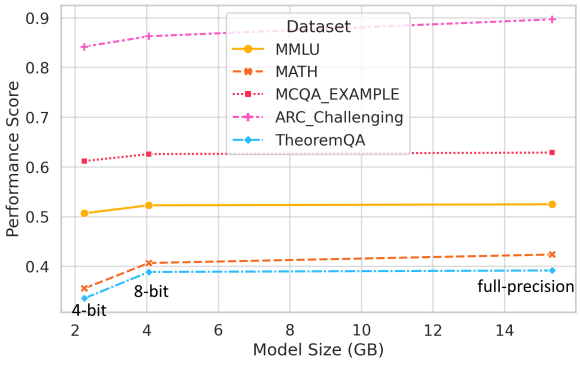


Figure 4: Difference in model performance with 4-bit, 8-bit, and full-precision on the different benchmark datasets.

## 5 Analysis

To evaluate the performance of our model qualitatively, we examine individual domains from the MMLU dataset. This approach allows us to identify patterns in the model's strengths and weaknesses. Although we can improve on the base model, the results in Table 3 still show relatively poor performance in fields such as mathematics and physics, where complex reasoning is more likely required. Conversely, the model performs exceptionally well in biology, which contains more factual, knowledge-based questions. Thus, whilst DPO seems like a promising method for improving complex reasoning, using higher-quality data could likely improve results further.

We also note that while ProbLLM-3.8B-RAG performs better in two domains when compared with Phi-3-mini-4k-instruct-RAG, the latter outperforms ProbLLM-3.8B-RAG in all domains where the data is included in the RAG index database. In other words, Phi-3-mini-4k-instruct-RAG excels in domains relevant to EPFL courses, for which we need to optimize our model. Furthermore, a qualitative analysis of incorrect answers chosen by ProbLLM-3.8B-RAG reveals that it often fails to utilize the available context (see Appendix F). We argue that DPO may impair the in-context learning capabilities of our model. This is further supported by the observation that zero-shot prompting is more effective than three-shot prompting variants for ProbLLM-3.8B, as shown in Table 5.

## 6 Ethical considerations

All of our training and benchmarking is completed using English datasets, however, it is important to consider that English is not the primary language spoken by most people, not only globally but even within the EPFL community. As such, to improve the accessibility of our AI tutor to ensure equality, it will be essential to adapt it to handle other languages. For high-resource languages, we could simply use multilingual base models and add education data from these languages in the finetuning process. For low-resource languages, where less high-quality data is available and for which pre-trained multilingual LLMs may underperform (Conneau et al., 2019), we would likely require additional dataset curation and training of language-specific adapters to improve performance (Pfeiffer et al., 2020).

In addition to spoken languages, adapting our model for sign language is essential for the inclusivity of the deaf community. This can be achieved by integrating sign language translation and generation techniques into the existing model (Bragg et al., 2019). The process involves translating a

Table 2: Performance evaluations of ProbLLM-3.8B models across different configurations. The values in the cells correspond to multiple-choice question correctness accuracies, and the cell with the highest accuracy among each datasets is in bold.

| Model | MMLU | MATH | ARC Challenging | TheoremQA | Overall |
|---|---|---|---|---|---|
| ProbLLM-3.8B | 0.525 | 0.424 | **0.897** | **0.392** | **0.559** |
| ProbLLM-3.8B-8bit | 0.523 | 0.407 | 0.863 | 0.389 | 0.546 |
| ProbLLM-3.8B-4bit | 0.507 | 0.356 | 0.842 | 0.336 | 0.510 |
| ProbLLM-3.8B-RAG | 0.511 | **0.457** | 0.834 | 0.339 | 0.535 |
| Phi-3-mini-4k-instruct-RAG | **0.526** | 0.452 | 0.843 | 0.342 | 0.541 |

Table 3: Performance evaluations of ProbLLM-3.8B models across different domains from MMLU. The values in the cells correspond to multiple-choice question correctness accuracies.

| Domain | ProbLLM-3.8B | ProbLLM-3.8B-RAG | Phi-3-mini-4k-instruct-RAG |
|---|---|---|---|
| College Mathematics | 0.302 | 0.345 | **0.414** |
| College Physics | **0.432** | 0.398 | **0.432** |
| College Chemistry | 0.480 | **0.530** | 0.480 |
| College Computer Science | 0.483 | 0.509 | **0.552** |
| Machine Learning | 0.500 | 0.446 | **0.563** |
| College Biology | **0.833** | 0.826 | 0.813 |

sign language query into English text, generating an answer, and then converting the answer back into sign language.

Whilst our model could significantly benefit students and educators if it performs as intended, it is currently trained on only a subset of STEM topics offered at EPFL (mathematics, physics, computer science, and electrical engineering). As a result, students from other disciplines might be excluded, potentially undermining equality and fairness. To address this issue, we should expand the training data to encompass all courses offered at EPFL.

Furthermore, AI tutors, while beneficial, pose risks of misuse. Students might use them to cheat on assignments, which undermines the educational process. Overreliance on the AI tutor could also deprive students of critical learning experiences and potentially mislead them if the model provides incorrect answers. Establishing strong guidelines and policies for responsible usage is imperative to mitigate these risks.

On a broader scale, despite their potential to enhance educational outcomes, AI tutors require significant computational resources, which can limit their accessibility. Running such models, even with optimizations, demands considerable compute power or stable internet connections, which may not be available in lower-resource settings. This limitation could exacerbate educational disparities between high- and low-socioeconomic communities. Mitigating this would likely require government intervention to ensure AI-based education is widely accessible.

## 7 Conclusion

In this work, we demonstrate several promising approaches for adapting general-purpose LLMs for STEM education, where the requirement of complex reasoning makes many current models insufficient. By using DPO, we can successfully align the model towards stronger reasoning capabilities for STEM topics. Additionally, we show that RAG can improve accuracy by integrating external knowledge, though our current method reveals that combining DPO and RAG may impair the model's in-context learning capabilities. This suggests that further refinement, such as incorporating RAG context during DPO training, could enhance performance. We also explore quantization techniques to reduce the computational footprint of our models, making them more accessible for deployment in resource-constrained environments. Despite a performance drop with quantization, the significant reduction in resource requirements is a crucial step toward the wider availability of AI-based education tools. Future work should expand training datasets to cover more courses and develop strategies to mitigate DPO's impact on in-context learning, moving closer to an effective AI tutor for diverse educational settings.

## References


Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*.

Yasmina Al Ghadban, Huiqi (Yvonne) Lu, Uday Adavi, Ankita Sharma, Sridevi Gara, Neelanjana Das, Bhaskar Kumar, Renu John, Praveen Devarsetty, and Jane E. Hirst. 2023. Transforming healthcare education: Harnessing large language models for frontline health worker capacity building using retrieval-augmented generation. *medRxiv*.

Ning Bian, Xianpei Han, Le Sun, Hongyu Lin, Yaojie Lu, Ben He, Shanshan Jiang, and Bin Dong. 2024. Chatgpt is a knowledgeable but inexperienced solver: An investigation of commonsense problem in large language models.

Danielle Bragg, Oscar Koller, Mary Bellard, Larwan Berke, Patrick Boudreault, Annelies Braffort, Naomi Caselli, Matt Huenerfauth, Hernisa Kacorri, Tessa Verhoef, et al. 2019. Sign language recognition, generation, and translation: An interdisciplinary perspective. In *Proceedings of the 21st International ACM SIGACCESS Conference on Computers and Accessibility*, pages 16–31.

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

Wenhu Chen, Ming Yin, Max Ku, Pan Lu, Yixin Wan, Xueguang Ma, Jianyu Xu, Xinyi Wang, and Tony Xia. 2023. Theoremqa: A theorem-driven question answering dataset.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05457v1*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

Tim Dettmers and Luke Zettlemoyer. 2023. The case for 4-bit precision: k-bit inference scaling laws. In *International Conference on Machine Learning*, pages 7750–7774. PMLR.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021a. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pages 2790–2799. PMLR.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Shijia Huang, Jianqiao Zhao, Yanyang Li, and Liwei Wang. 2023. Learning preference model for LLMs via automatic preference data generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9187–9199, Singapore. Association for Computational Linguistics.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. 2018. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2704–2713.

Selçuk Kılınç. 2023. Embracing the future of distance science education: Opportunities and challenges of chatgpt integration.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Nathan Lambert, Lewis Tunstall, Nazneen Rajani, and Tristan Thrush. 2023. Huggingface h4 stack exchange preference dataset.

Zachary Levonian, Chenglu Li, Wangda Zhu, Anoushka Gade, Owen Henkel, Millie-Ellen Postle, and Wanli Xing. 2023. Retrieval-augmented generation to improve math question-answering: Trade-offs between groundedness and human preference.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: Communicative agents for "mind" exploration of large language model society.

Jerry Liu. 2022. LlamaIndex.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. 2024. An empirical study of catastrophic forgetting in large language models during continual fine-tuning.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. 2022. Peft: State-of-the-art parameter-efficient fine-tuning methods. https://github.com/huggingface/peft.

Mujtaba Mateen. 2023. Computer science theory qa dataset.

R Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. *arXiv preprint arXiv:1902.01007*.

Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. 2024. A comprehensive overview of large language models.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020. Mad-x: An adapter-based framework for multi-task cross-lingual transfer. *arXiv preprint arXiv:2005.00052*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Rafael Rafailov, Joey Hejna, Ryan Park, and Chelsea Finn. 2024a. From r to q*: Your language model is secretly a q-function. *arXiv preprint arXiv:2404.12358*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024b. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, and Shengyi Huang. 2020. Trl: Transformer reinforcement learning. https://github.com/huggingface/trl.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. 2023a. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. 2023b. C-pack: Packaged resources to advance general chinese embedding.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2022. An explanation of in-context learning as implicit bayesian inference.

## A Corruption prompts

In Table 4, we provide the specific prompts used to corrupt answers for generating the preference pairs as described in (Huang et al., 2023). These prompts enable the transformation from correct to rejected answers through deletion, substitution, and insertion of content.

| Corruption Type | Example |
|---|---|
| **Deletion** | You are an adversarial actor trying to corrupt the correct answers. Your goal is to REMOVE some answer content that is useful to the question. |
| **Substitution** | You are an adversarial actor trying to corrupt the correct answers. Your goal is to EDIT some parts of the answer content to make it INACCURATE for the question. |
| **Insertion** | You are an adversarial actor trying to corrupt the correct answers. Your goal is to ADD some answer content irrelevant to the question into the answer. |

Table 4: Detailed Examples of Corruption Prompts

## B MMLU Topics

We retain questions from the following subjects in the MMLU dataset, organised by topic and with their respective question counts:

- **Math:**
  - Abstract Algebra (116)
  - Elementary Mathematics (424)
  - College Mathematics (116)
  - Formal Logic (145)
  - High School Mathematics (304)
  - High School Statistics (244)
- **Physics:**
  - Astronomy (173)
  - College Chemistry (113)
  - Conceptual Physics (266)
  - College Physics (118)
  - High School Physics (173)
- **Computer Science:**
  - College Computer Science (116)
  - High School Computer Science (114)
  - Machine Learning (128)
  - Computer Security (116)
- **Engineering:**
  - Electrical Engineering (166)

## C Documents Collection for RAG

We provide a comprehensive list of all EPFL courses for which we have identified relevant documentation sources (course bibliographies or lecture notes) for our RAG knowledge base. Courses were selected based on their alignment with topics in the preference pairs dataset. It is important to note that not every EPFL course from the respective categories is included; many courses were omitted due to overlapping bibliographies, which made additional documentation redundant.

- **Electrical and Electronics Engineering (EL)**
  - EE-452 Network machine learning
  - EE-559 Deep learning
  - EE-566 Adaptation and learning
- **Computer Science (IN)**
  - CS-200 Computer architecture
  - CS-202 Computer systems
  - CS-206 Parallelism and concurrency
  - CS-250 Algorithms I
  - CS-401 Applied data analysis
  - CS-412 Software security
  - CS-431 Introduction to natural language processing
  - CS-433 Machine learning
  - CS-450 Algorithms II
  - CS-452 Foundations of software
  - CS-456 Artificial neural networks/reinforcement learning
  - CS-460 Systems for data management and data science
  - CS-503 Visual intelligence: machines and minds
  - CS-526 Learning theory
  - CS-550 Formal verification
  - CS-552 Modern natural language processing
- **Physics (PH)**
  - PHYS-114 General physics: electromagnetism
  - PHYS-200 Physics III
  - PHYS-207 Quantum mechanics I
  - PHYS-313 Quantum physics I
  - PHYS-324 Classical electrodynamics
  - PHYS-415 Particle physics I
  - PHYS-425 Quantum physics III
  - PHYS-431 Quantum field theory

## D Additional Experiments

### D.1 Answer Extraction and Prompting Techniques

#### D.1.1 Zero-shot prompting

We utilize the following zero-shot prompt:

*"Start by saying the letter corresponding to the correct answer (A, B, C, or D), and include your reasoning after.*

*Question: [Text of the question]*

*Answer:"*

### D.1.2 3-shot prompting

We utilize the following 3-shot prompt:

*"Start by saying the letter corresponding to the correct answer (A, B, C, or D), and include your reasoning after.*
*You are a specialist at solving engineering-related questions and are rewarded handsomely for each correct answer. You will be given multiple choice questions that you have to answer, which you will answer with a single character indicating your choice. The following are some examples of the expected input and output:*

*Question: Statement 1| Linear regression estimator has the smallest variance among all unbiased estimators. Statement 2| The coefficients $\alpha$ assigned to the classifiers assembled by AdaBoost are always non-negative.*

*Options:*
*A. True, True*
*B. False, False*
*C. True, False*
*D. False, True*

*Answer: D*

*Question: A rise in intracellular free calcium in the sea urchin oocyte causes the release of proteolytic enzymes which act to prevent polyspermy. The events just described entail the?*

*Options:*
*A. zona reaction*
*B. acrosomal reaction*
*C. cortical reaction*
*D. fertilization reaction*

*Answer: C*

*Question: Of the following atoms, which has the lowest electron affinity?*

*Options:*
*A. F*
*B. Si*
*C. O*
*D. Ca*

*Answer: D*

*Question: [Text of the question]*

*Answer:"*

### D.1.3 3-shot prompting + reasoning

We utilize the following 3-shot prompt with reasoning:

*"You are a specialist at solving engineering-related questions and are rewarded handsomely for each correct answer. You will be given multiple choice questions that you have to answer, which you will answer with a single character indicating your choice. The following are some examples of the expected input and output:*

*Question: Statement 1| Linear regression estimator has the smallest variance among all unbiased estimators. Statement 2| The coefficients $\alpha$ assigned to the classifiers assembled by AdaBoost are always non-negative.*

*Options:*
*A. True, True*
*B. False, False*
*C. True, False*
*D. False, True*

*Answer:D*

*Reasoning: Statement 1 is false because only linear estimators are considered in the Gauss-Markov theorem. Statement 2 is true as coefficients in AdaBoost are non-negative.*

*Question: A rise in intracellular free calcium in the sea urchin oocyte causes the release of proteolytic enzymes which act to prevent polyspermy. The events just described entail the?*

*Options:*
*A. zona reaction*
*B. acrosomal reaction*
*C. cortical reaction*
*D. fertilization reaction*

*Answer:C*

*Reasoning:The cortical reaction releases enzymes to prevent multiple sperm from fertilizing the egg, thus preventing polyspermy.*

*Question: Of the following atoms, which has the lowest electron affinity?*

*Options:*
*A. F*
*B. Si*
*C. O*
*D. Ca*

*Answer:D*

*Reasoning:Ca, being a metal, typically has lower electron affinity compared to non-metals like F, Si, and O.*

*Question: [Text of the question]*

*Answer:"*

### D.2 Zero-shot prompting + RAG

We utilize the following prompting scheme for RAG:

> *"Start by saying the letter corresponding to the correct answer (A, B, C, or D), and include your reasoning afterwards. You can use the contextual knowledge we provide to help you answer the question.*
>
> *Contextual knowledge 1: [Text of context]*
>
> *Contextual knowledge 2: [Text of context]*
>
> *Contextual knowledge . . . : [Text of context]*
>
> *Contextual knowledge $n$: [Text of context]*
>
> *Question: [Text of the question]*
>
> *Answer:"*

### D.3 Prompting Techniques and Answer Extraction

Table 5: Multiple-choice question answering accuracy for different prompting techniques (few-shot $\pm$ reasoning) and answer extraction schemes (MCQ-Logits or not). The bold cells attain the highest performance among models on the corresponding dataset.

| Prompting Technique | MMLU | MATH | ARC Challenging | TheoremQA |
|---|---|---|---|---|
| 0-shot | 0.517 | 0.329 | 0.852 | 0.363 |
| 3-shot | 0.541 | 0.307 | 0.873 | 0.385 |
| 3-shot reasoning | **0.543** | 0.332 | **0.874** | 0.363 |
| 0-shot MCQ-Logits | 0.525 | **0.424** | 0.867 | **0.392** |
| 3-shot MCQ-Logits | 0.541 | 0.308 | 0.872 | 0.378 |
| 3-shot reasoning MCQ-Logits | 0.537 | 0.332 | **0.874** | 0.373 |

### D.4 Quantization

Table 6: Performance of models across different quantization levels

| Quantization | Model Size (Bytes) | MMLU | MATH | ARC Challenging | TheoremQA |
|---|---|---|---|---|---|
| Full-precision | 15334649856 | 0.525 | 0.424 | 0.897 | 0.392 |
| 8-bit | 4068612096 | 0.523 | 0.407 | 0.863 | 0.389 |
| 4-bit | 2256627268 | 0.507 | 0.356 | 0.842 | 0.336 |

## E Training Hyperparameters

| Parameter | Value |
|---|---|
| Seed | 0 |

| Parameter | Value |
|---|---|
| Precision | bfloat16 |
| Optimizer | paged_adamw_32bit |
| Top_k | 50 |
| Top_p | 1 |
| Steps | 2000 |
| Epochs | 1.533 |
| Adam_beta1 | 0.9 |
| Adam_beta2 | 0.999 |
| Eval_steps | 100 |
| Hidden_act | silu |
| Max_generation_length | 1024 |
| Vocab_size | 102400 |
| Adam_epsilon | 1e-8 |
| Learning_rate | 1e-5 |
| Learning_rate_schedule | cosine |
| Warmup_ratio | 0 |
| Warmup_steps | 100 |
| Weight_decay | 0.05 |
| Max_grad_norm | 1 |
| DPO_beta | 0.05 |
| Lora_alpha | 16 |
| Lora_r | 8 |
| Lora_dropout | 0.05 |
| Batch_size | 8 |

## F RAG Failure Cases

We now show three examples where the RAG model could have utilized its context to solve the given question, but failed to do so.

**Question 1**

**Prompt:** Start by saying the letter corresponding to the correct answer (A, B, C, or D), and include your reasoning afterwards. You can use the contextual knowledge we provide to help you answer the question.

**Contextual knowledge 1:** Figure 10.10 describes the architecture of a convolutional network model, specifically VGG-16, which features standard neural network layers with full connectivity and no parameter sharing. The final max-pooling layer comprises 512 channels, each of size $7 \times 7$, totaling 25,088 units. This is followed by a series of fully connected layers with 4,096 units each, and a final layer consisting of 1,000 units tailored for a classification task involving 1,000 different classes. All layers, except for the output layer which utilizes a softmax activation function, employ nonlinear ReLU activations. VGG-16 has approximately 138 million learnable parameters, with the majority (nearly 103 million) in the first fully connected layer.

**Contextual knowledge 2:** The described architecture employs bottleneck residual blocks instead of traditional convolutional layers, similar to AlexNet and VGG. These blocks are interspersed with periods of decreasing spatial resolution and increasing channel numbers, achieved through downsampling via stride two convolutions and enhancements via 1x1 convolutions or zero-padding. The network starts with a 7x7 convolutional layer, proceeds through downsampling, and concludes with a fully connected layer that outputs a 1000-length vector, processed by a softmax layer for class probability estimation. The ResNet-200 model, using this architecture, achieved a top-five error rate of 4.8% and a top-one error rate of 20.1%, surpassing human performance benchmarks at the time of its conception in 2016.

**Contextual knowledge 3:** Graph neural networks integrate many existing graph algorithms into a cohesive framework, with significant applications in areas like graph and node classification, edge

prediction, and graph clustering. Introduced by Gori et al. (2005) and further developed by Scarselli et al. (2008), these networks update node embeddings iteratively through a contraction mapping function, incorporating both node and edge information from the network's graph structure.

**Question:** As of 2020, which architecture is best for classifying high-resolution images?

**Options:**

- A. convolutional networks
- B. graph networks
- C. fully connected networks
- D. RBF networks

**Answer:** Actual: A Pred: B

**Question 2**

**Prompt:** Start by saying the letter corresponding to the correct answer (A, B, C, or D), and include your reasoning afterwards. You can use the contextual knowledge we provide to help you answer the question.

**Contextual knowledge 1:** Decision Trees involve a representation where each block can be described using $\log_2(d+3)$ bits, assuming each internal node branches into two children. The encoding of a decision tree with $n$ nodes can be described by a length of $(n+1)\log_2(d+3)$. By Theorem 7.7, for any decision tree $h \in H$ with $n$ nodes, sampled over size $m$, the true risk $LD(h)$ is bounded as:

$$LD(h) \leq LS(h) + \sqrt{\frac{(n+1)\log_2(d+3) + \log(2/\delta)}{2m}} \quad (18.1)$$

This suggests a trade-off between the complexity of the tree and its empirical risk $LS(h)$. The goal is to find a tree that balances low empirical risk with a manageable number of nodes.

**Contextual knowledge 2:** Decision Tree Algorithms utilize Equation (18.1) to derive a learning rule, which suggests searching for a tree that minimizes the right-hand side of the equation. However, this problem is computationally challenging. Practical decision tree algorithms, therefore, rely on heuristics like greedy approaches where decisions are made locally at each node construction. The growth of a decision tree starts with a root and iteratively splits leaves to maximize a defined "gain" measure, typically choosing the split that maximizes gain or opting not to split at all.

**Contextual knowledge 3:** Randomized Decision Trees consider a model where branches are determined either by a deterministic process or a random decision, focusing on distributions over deterministic trees. For an input $x$, the expected number of queries a tree makes is denoted by $c(P, x)$. The randomized decision tree complexity, $R(f)$, is defined as:

$$R(f) = \min_{P} \max_{x} c(P, x) \quad (7)$$

This complexity metric evaluates how effective the best possible tree distribution performs against the worst possible input.

**Question:** Evaluate the following statements:

1. For a continuous random variable $x$ and its probability distribution function $p(x)$, it holds that $0 \leq p(x) \leq 1$ for all $x$.
2. Decision tree learning is driven by minimizing information gain.

**Options:**

- A. True, True
- B. False, False

- C. True, False
- D. False, True

**Answer:** Actual: B Pred: A

**Question 3**

**Prompt:** Start by saying the letter corresponding to the correct answer (A, B, C, or D), and include your reasoning afterwards. You can use the contextual knowledge we provide to help you answer the question.

**Contextual knowledge 1: Supervised Learning Training and Testing**

- Training involves finding parameters that minimize the loss, termed as model fitting. This process usually starts with random parameter values and improves by gradient descent until no further improvements can be made.
- Testing evaluates how the model performs on new, unseen data. The performance depends partly on the training data's representativeness and the model's expressiveness. Overfitting and underfitting are critical issues affecting model performance.

**Contextual knowledge 2: Generalization Theory**

- Overly complex models can overfit, performing well on training data but poorly on new data, while overly simple models might underfit, failing to capture the underlying pattern of the data.
- The bias-variance tradeoff is a fundamental concept in model design, aiming to balance simplicity and complexity to minimize overall error.

**Contextual knowledge 3: Model Selection and Validation**

- Model selection involves choosing the best model and setting its parameters based on how well it fits the training data without overfitting.
- For instance, choosing the degree of a polynomial in regression affects whether the model will underfit or overfit the data.

**Question:** _refers to a model that can neither model the training data nor generalize to new data.
**Options:**

- A. good fitting
- B. overfitting
- C. underfitting
- D. all of the above

**Answer:** Actual: C Pred: A

## G Contribution Statement

All four of us wrote the reports. Overall, we contributed to the project equally.