From Questions to Calculus: A multisystem AI Assistant for STEM Success

Madeleine Hueber | 389346 | madeleine.hueber@epfl.ch
Tim Kluser | 394411 | tim.kluser@epfl.ch
Michelle Odnert | 389525| michelle.odnert@epfl.ch
Flavia Wallenhorst | 264996 | flavia.wallenhorst@epfl.ch

Abstract

Language models have recently shown great promise for education, yet aligning them with human preferences while ensuring efficiency remains challenging. In this work, we introduce a STEM-focused AI assistant composed of four specialized components: a reward model, a MCQA model, a retrieval-augmented generation (RAG) model, and a quantized inference system. Each is finetuned from the Qwen3-0.6B-Base language model using carefully curated datasets and improved with advanced techniques such as curriculum learning or QLoRA. Our experiments demonstrate that our models achieve promising results on benchmark test set highlighting the model's potential as an assistive tool in educational settings.

Introduction

Recent breakthroughs in large language models have made automated tutoring possible, yet turning these systems into reliable, classroom-ready mentors is still far from trivial. Existing methods often overfit narrow benchmarks or require computational budgets beyond the reach of most schools. In this project, we explored four modeling strategies to enhance an AI assistant's performance: reward alignment, MCQA training, retrieval-augmented generation, and quantized inference.

1 Approach

1.1 Reward model

The reward model was developed to assess the quality of model-generated responses, thereby our goal was to fine-tune a language model to assign higher rewards to outputs that are more pedagogically valuable or accurate in STEM contexts.

We built the model in two stages. First, we fine-tuned the Qwen3-0.6B-Base model on a curated instruction-following dataset focused on STEM topics to instill domain-specific capabilities. Then we applied Direct Preference Optimization (DPO) (Rafailov et al., 2024) to further align the model with human preferences. To enhance performance, we explored two complementary directions: curriculum learning and alternative loss functions.

Curriculum learning

To improve the model's ability to capture nuanced preferences, we adopted a curriculum learning strategy (Pattnaik et al., 2024).The preference dataset was divided into three levels—easy, medium, and hard—based on the score gap between answer pairs. Training proceeded in stages over three epochs: starting with easy pairs, then adding medium ones, and finally including the full dataset.

Loss functions

To increase robustness, we explored variants of the DPO loss: the standard DPO (Rafailov et al., 2024), IPO (Implicit Preference Optimization) (Azar et al., 2023), which reduces overfitting by regulating log-likelihood gaps, and Robust-DPO (Chowdhury et al., 2024), which explicitly models label noise. Full details and mathematical formulations are provided in the Appendix A.1. These alternatives can be especially useful given the inherent variability in human preference data.

1.2 Quantized Model

We begin our quantization experiments using our best MCQA model as a baseline, focusing on weight quantization with LLM.int8() techniques introduced by Dettmers et al. (Dettmers et al., 2022) and implemented in the BitsAndBytes library.

BitsAndBytes implements a vector-wise quantization scheme with adaptive outlier handling, designed to preserve model accuracy even under aggressive compression. Our approach only make use of the grouped quantization as, according to Dettmers et al. (Dettmers et al., 2022), adaptive outlier handling benefits are minimal for models under 6.7B parameters, especially below 1B.

Additionally, we explore the effects of parameter-efficient fine-tuning using Low-Rank

Adapters (LoRA) (Dettmers et al., 2023). In this setup, the original model parameters are frozen, and a small number of trainable parameters are added to the model. This enables task adaptation with low additional computational overhead.

1.3 Distillation inspired RAG

To improve our MCQA model, we built a RAG model. While not pure distillation, we decided to distill knowledge from GPT-4o-mini and GPT-4.1-mini for our RAG documents. We prompted the teacher model with multiple-choice questions and answers from four different MCQA datasets asking for relevant information regarding the topic. For further improvement, we fintuned an embedding model using the same data.

2 Experiments

2.1 Data

2.1.1 MCQA, Quantization, and base RAG

The multiple choice model is fine-tuned on a simple dataset consisting of OpenAI's ai2-arc easy training data, preference pairs crowd-sourced from this course (CS-552), and GSM8k training data.

ai2-arc: The ai2-arc easy training data is a mcqa dataset that features STEM problems from the high school and early college level. (Clark et al., 2018)

preference pairs: The preference pairs consists of highly technical college-level STEM questions. In the final model, the selected preference pairs were filtered down to only mcqa questions with 4 options, where the question length is less than 30 characters. The character limit was put in place to simulate an 'easy' subset of the preference pairs.

gsm8k: The GSM8k data is a well-known math benchmark. (Cobbe et al., 2021) The publicly available training data is short answer. To use it for our MCQA task, we created a method to automatically generate 3 wrong answers using the chat-gpt wrapper from M1 (preference pairs). The three wrong answers for every existing question/answer pair should not deviate too much from the correct answer but mutually deviate enough.

Despite the instruction to not repeat the question and directly give the answer, there was still a lot of noise in the data which required manual cleaning. In the end, we could use 70% of the generated samples. The prompt we used to generate the pairs is detailed in Appendix A.12.

other data: There were 26000 rows of data that we trained and evaluated on, but ultimately did not improve the performance of this model.

The final model was trained on 3000 rows. All the data used for this project was public and widely used STEM datasets, meaning we incurred little legal risk. We acknowledge the ethical consideration that the dataset is entirely in English and therefore is not providing a global perspective.

2.1.2 RAG Documents

For our RAG documents, we utilized different STEM datasets described in Table 4, The (1) SciQ dataset is a multiple-choice question answering (MCQA) resource consisting of questions generated by crowd workers based on textbook materials in STEM subjects for students in grades 4 through 8, referred to as support content. This support content was pre-classified by the original authors.

We sought to replicate their methodology for support creation on EPFL course materials, applying a Document Filter incorporating lexical, grammatical, pragmatic, and complexity-based criteria. However, we observed that without human-driven post-processing such as the classification performed by crowd workers in SciQ the resulting data quality remained insufficient.

Alternatively, we generate synthetic data based on our four datasets to replicate the support column from the SciQ dataset. The (2) StemQ dataset is a university-level MCQA dataset focused on STEM disciplines — the questions are typically multi-hop in nature and often require specific calculations. We also used a subset of the (3) AI2-ARC challenging dataset and a subset of (4) preference pairs crowd-sourced from the CS-552 course.

2.1.3 Reward model

We used two datasets aligned with the model's development stages: instruction tuning and DPO.
For Instruction Tuning, we used the TIGER-Lab/WebInstruct-verified dataset (Ma et al., 2025), which provides high-quality STEM content across academic levels. Its coverage and reliability made it ideal for teaching our model to follow educational instructions effectively.
For DPO, we built a custom dataset combining STEM-focused pairs from multiple sources:

- A subset of Milestone 1, with one pair per question to reduce annotation inconsistencies.
- STEM-related examples from argilla/Capybara-Preferences, distilled from high-quality reasoning datasets including GOAT, Verified-CAMEL,

and TheoremQA (used only for evaluation).

- argilla/distilabel-math-preference-dpo dataset.
- A STEM-focused subset of the human-annotated allenai/multipref dataset (Miranda et al., 2025) .

The final dataset was split into train, validation, and test sets; the first two were used for DPO fine-tuning, and the test set reserved for evaluation.

2.2 Evaluation method:

MCQA, Quantization, and RAG

For evaluation of our MCQA model we use the following datasets that show how good our model performs on different tasks. While the general STEM Problem Set consists of very broad and diverse knowledge in STEM subjects of different difficulties, the gsm8k is more math, logics and syntax heavy.

- GSM8k Test Set (Cobbe et al., 2021): Using the test set provided, we again used the GPT wrapper to develop our own mcqa test set for GSM8k to measure our performance on math-related tasks. Total of 378 rows.
- General STEM Problem Set: We sombined some MMLU questions, and the arc-easy and arc-challenge test sets. Total of 3722 rows.
- zechen-nlp/MNLP_STEM_mcqa_evals: A 770 question general STEM dataset provided by EPFL CS-552 teaching team.

Reward model

We evaluated our reward model with reward accuracy on the held-out test split of our custom preference dataset. Additionally, we also evaluated on the zechen-nlp/MNLP_dpo_evals dataset as an external benchmark. This allowed us to evaluate how our model aligned with the course's expectations.

2.3 Experimental details:

All models were trained on one NVIDIA GPU. Hyperparameter details can be found in the Appendix.

2.3.1 MCQA

To train the MCQA model, we employed supervised fine-tuning (SFT) using both one-round and multi-round methods. The one-round method trained the model on all data simultaneously, while the multi-round method adopted curriculum training, progressively increasing difficulty. This involved first training on an 'easy' set (ai2_arc easy, short preference pairs, and some GSM8k questions), followed by a 'medium' set (MMLU questions under 302 tokens), and finally a 'challenging' set (ai2_arc challenge and longer preference pairs).

2.3.2 Quantized Model

4-bit: We load model weights in 4-bit using NormalFloat4 (NF4), a non-uniform quantization that better preserves performance than uniform methods like FP4 or int4. We further use double quantization, to further improve memory efficiency and mitigate the accuracy drop often associated with aggressive quantization. For inference, we convert the weights back to 16 bits.

8-bit: We load model weights in 8-bit using LLM.int8(), which applies vector-wise quantization and keeps high-precision outlier weights. As explained in the Approach section, the hybrid quantization has no impact on our small model.

QLoRA adapter: We fine-tune a 4-bit quantized model using QLoRA (Dettmers et al., 2023), adding trainable Low-Rank Adapters on frozen weights. We target the attention projection layers and apply a dropout of 0.05. Only LoRA weights are updated on a subset of the training data of the base model.

2.3.3 Reward model

We experimented with three loss functions during DPO training: standard DPO, Robust DPO (with 1e-4 label error probability), and IPO . However, IPO caused exploding gradients that we were unable to resolve, so it was ultimately discarded.

2.3.4 RAG

We used subsets of our datasets specified in Table 1 to facilitate comparison with the benchmark, balance dataset impact, and perform data cleaning when necessary.

For the distillation, we isolated the questions and full-text answers from subsets of our four datasets: SciQ, StemQ, ARC2AI, and the preference pairs. We then explored various prompts, always including both the questions and the answers, until we found one that produced satisfactory results (see Appendix). In particular, we experimented with generating university-level documents based on college-level MCQA by specifying in the prompt that the content should be at the master's level. We didn't perform any chunking as we specified in our prompt to generate concise response. Embedding model fine-tuning was performed using questions

and answers from the SciQ dataset, applying a positive pair approach. We used a domain-specific embedding model pre-trained on medical data, as it yielded the best performance on our tasks and medicine is considered a subdomain of STEM.

2.4 Results:

We evaluated our four components against relevant baselines to assess their performance improvements. For MCQA, RAG, an the quantized model we use **Qwen3-0.6B-Base** as a primary baseline. For RAG and the quantized model we also compared against the best-performing full-precision MCQA model. The reward model is compared against **Qwen3-0.6B**, a fine-tuned version of the base model.

2.4.1 MCQA, Quantized Model and RAG

Model	Broad	MCQA Evals	GSM8K / GPT
Qwen3-0.6B-Base	0.7507	0.4247	0.4312
echallenge_mcqa	0.7168	0.3800	0.5566
mcqa	0.7507	0.4169	0.5635
4bit quant	0.6932	0.3610	0.5582
4bit quant**	0.6937	0.3636	0.5635
4bit quant + LoRA	0.6926	0.3597	0.5582
8bit	0.7184	0.3922	0.5688
Qwen3-0.6B-Base RAG	0.7499	0.4299	0.3889
echallenge_mcqa RAG	0.6975	0.3403	0.5423
mcqa RAG	0.7437	0.3896	0.4392

Table 1: Normalized accuracy on selected datasets. **4-bit without double quantization.`echallenge`: curriculum-trained MCQA; `mcqa`: trained on full data.

Model size	Memory (GB)	Peak Mem. (GB)	Peak Mem. Long* (GB)
Qwen3-0.6B-Base	2.221	2.249	2.370
8-bit Quantization	0.735	0.820	0.970
4-bit Quantization	0.539	0.523	0.583
4-bit + LoRA	0.548	0.532	0.592
4-bit**	0.7	0.542	0.602

Table 2: Model size and peak memory usage during inference. *Long prompt = 10× original prompt length. **4 bit quantization in the same setting but without double quantizaion.

Table 1 reports accuracy for different datasets on our different models, while Table 2 details memory usage for quantized versions. The results show that 8-bit quantization offers strong performance with significant memory savings, but looking at the grading formula for quantization (accuracy / average peak ram), the normal 4 bit quantization offers the best overall trade-off. In comparison LoRA fine-tuning and 4 bit model without double quantization introduces an increase in memory usage without a clear gain. This justifies our choice of the 4-bit model as the final system.

Concerning our RAG model it did not show significant improvements across most of our evaluation benchmarks, regardless of the base model used. It only yielded a slight performance gain in the MCQA evaluation dataset. We provided an explanation for these results in the analysis section.

2.4.2 Reward model

We evaluated four model variants: DPO, DPO with curriculum learning (DPO-C), with robust loss (DPO-R), and with both (DPO-C-R). Their performance on our evaluation sets is reported in Table 3.

Model	**Custom test set**	**MNLP_dpo_evals**
`Qwen3-0.6B`	0.586 ± 0.013	0.566 ± 0.022
`Dpo`	0.693 ± 0.012	0.784 ± 0.018
`Dpo-C`	0.660 ± 0.013	$\mathbf{0.809 \pm 0.017}$
`DPO-R`	$\mathbf{0.694 \pm 0.012}$	0.795 ± 0.018
`Dpo-C-R`	0.602 ± 0.013	0.689 ± 0.020

Table 3: Reward accuracy for each model on our two evaluation sets.

As shown in Table 3, all four models outperform the Qwen3-0.6B baseline.Compared to DPO, curriculum learning (DPO-C) yields the best accuracy on the MNLP_dpo_evals set but does not improve performance on the custom test set. Robust loss (DPO-R) improves results on both and achieves the highest accuracy on the custom test set. However, combining both (DPO-C-R) leads to lower performance across the board. Based on these results, wselected DPO-R as our final model for its consistent performance.

3 Analysis

3.1 MCQA and Curriculum Training

Our results show that curriculum training was not effective for this task. Figure 1 displays the accuracy at each stage of training on every test set (results are from klusertim/MNLP_M3_mcqa_dataset 'broad' evaluation)

The accuracy is not correlated with our labeling of easy, medium, and challenging; and our best results come from training on all the data all at once. Given that our best outcome was achieved by training on the full dataset simultaneously, it

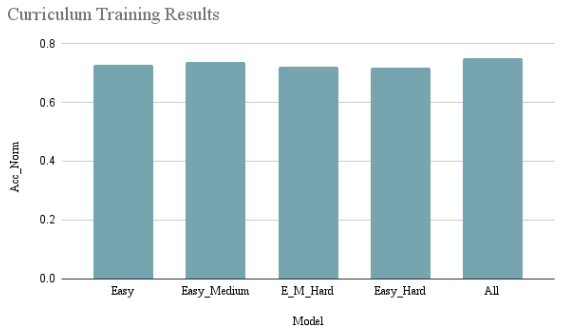

Figure 1: Curriculum training Accuracy. E_M_Hard is the model trained first on easy data, then medium, then hard.

follows that our difficulty labeling may not reflect the model's processing challenges, despite our labeling following the common standards of large benchmarks such as ai2_arc and mmlu.

Initially our model achieved reasonable performance on general understanding questions in STEM but struggled with precision-intensive questions such as in math questions. To understand the models capacities, we split our test data into more "syntax and math"-heavy questions and broad knowledge questions, along with more math questions in our training from the gms8k dataset. While performance on math improved, it still lagged behind broad STEM questions. Increasing math samples further (around 3000 examples) caused overfitting, lowering broad STEM and overall MCQA scores.

Efforts to enhance performance on broader knowledge by incorporating MMLU training data did not yield improvements. We attribute this to the complexity of the available auxiliary training data compared to the actual MMLU test questions.

3.1.1 Quantized model

For our quantized models with post training quantization techniques, we see that the performance is similar to our best mcqa model on all the topics. Interestingly, the quantized model is even a little bit better than our best mcqa model on the math questions. One explanation might be that our best model slightly overfits and a reduced version of it gets better in generalizing.

3.1.2 Different evaluation-suite versions

On the old version of the evaluation framework lighteval, our best mcqa model outperformed the base model by over 10% accuracy on our test data, scoring above 50% on the `mcqa evals` and `gsm8k` and 70% on `broad`. Inexplicably, that changed after the lighteval update, despite using the same single-token continuation task. On the new version, our best models performance decreased and now match the base model. This may be due to changes in the environment, as results remained low even when running the old Lighteval with the updated transformers/torch versions.

3.2 RAG Analysis

When testing with extended or shortened version of the SciQ dataset, we mistakenly attributed poor performance to input length instead of realizing that the dataset itself was ill-suited for our evaluation. (See Table 9) This led to overfitting to a small, narrow task, which explains why our RAG pipeline shows limited gains on broader benchmarks. Likewise, constructing RAG documents solely from the SciQ support corpus failed to boost performance, however adding our three synthetic training sets produced incremental gains relative to Qwen3-0.6B-Base as shown in Figure 3.

Data Generation May Outperform Textbook Content: Having at least obtained convincing results by generating synthetic data from our MCQA datasets, we applied the same process to the SciQ dataset. We compared the carefully selected textbook passages to a generated version using GPT-4.1 Mini. The generated data outperformed the textbook data as shown in Figure 4.

Importance of the GPT Model: Given that crowd workers are expensive, openly available textbooks are limited, and the process of classifying textbook passages is quite challenging, it is important to determine the minimum performance and therefore cost required for generated data to surpass textbook quality. For our generated datasets based on StemQ, ARC2-AI, and the preference pairs, we compared the results using GPT-4o Mini, which costs half as much as GPT-4.1 Mini (0.15 per million tokens), accompanied by the StemQ support dataset.We observe that while two times cheaper, the performance remains similar as shown in Figure 5.

Domain-Specific Embedding Model: We explored different embedding models : domain-specific MedEmbed-small-v0.1 performed best. However, the evaluation dataset we used contained a large proportion of medical-related questions. For this reason, we fine-tuned our own embedding model on the SciQ dataset to make it more versatile.

Results can be found on in Table 10.

3.3 Reward model

The reward model's training begins with instruction fine-tuning. Skipping this step led to a sharp performance drop (0.44 ± 0.01 on our test set, 0.47 ± 0.02 on MNLP_dpo_evals), highlighting its importance for domain adaptation. The second training step uses datasets spanning STEM topics and varying preference difficulty to enhance generalization across domains.

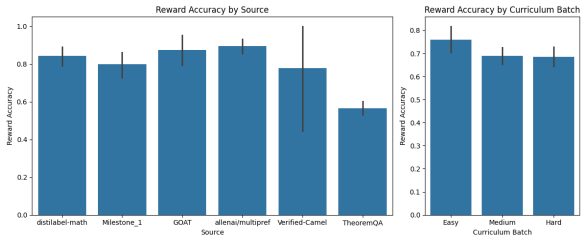

Figure 2: Reward accuracy across dataset sources (left) and curriculum batches (right) on our custom test set.

As Figure 2 shows, DPO-R performs best on allenai/multipref and GOAT. This is likely due to clearer preference contrasts and alignment with training data. GOAT in particular features clearer contrasts, such as obvious errors, unlike more subjective judgments based on style or fluency. The right panel confirms better performance on "easy" curriculum batches, consistent with expectations.

Metadata from the multipref dataset (further analyzed in the Appendix A.5) reveals higher accuracy on clearly defined preferences (e.g., CloseQA) and lower accuracy for style-based or subjective ones. Performance also drops in biology and climate science, likely due to math-heavy training data and domain-specific expectations.

3.4 Limitations and reflections

Building a high-quality dataset proved harder than expected. Limited and homogeneous data hurt generalization, and memory constraints on Noto forced us to drop long questions and keep validation small, likely reducing performance on longer inputs. Misled by early results, we built a very small RAG system that worked in narrow setups but failed to generalize—scaling it up would likely have helped. Finally, working in parallel introduced component dependencies we didn't fully account for, and we overlooked key intermediate results.

4 Ethical considerations

Adapting our Model to Other Languages
To extend our MCQA model to other high-resource languages (e.g., French), adaptation involves replacing the data with well-annotated MCQA datasets in the target language. Large pre-trained multilingual models (such as mBERT or Qwen-multilingual variants) can be used as the base, benefiting from prior exposure to these languages during pretraining. For low-resource languages (e.g Urdu), cross-lingual transfer learning and data augmentation can help overcome limited labeled data.
Beneficiaries : If the model functions as designed, all educational actors (teachers, students etc.) can benefit from automated, scalable, and consistent STEM assistant, improving access to high-quality practice and feedback.
Potential Harms : Trained solely on English data, the model may exclude non-English speakers and reflect Western biases, especially in DPO and RAG models that embed specific reasoning styles. Additionally, if the model makes mistakes, students might learn incorrect information, which could negatively impact their understanding.
Mitigation Measures : Potential harms can be reduced by ensuring transparency and explainability, so users understand the model's answers and scoring. Deployment should be limited to validated contexts with clear documentation of boundaries (e.g., excluding non-STEM domains). Additionally, diversifying datasets to better represent underrepresented groups and languages is essential.
Protecting Minority Users Marginalized groups may face disproportionate harm. To minimize this risk, it's essential to include diverse perspectives in model development and datasets.

5 Conclusion

This project demonstrates that even a compact 0.6B-parameter model with a targeted specialization pipeline, can effectively support a STEM-focused AI assistant. With curated data, tailored loss functions, and domain-specific tuning, promising performance can be achieved on structured STEM tasks. However, improvements are modest on broader benchmarks, and RAG-based gains diminish outside narrow domains— due to English-only data and memory constraints. Addressing these constraints through corpus scaling, multilingual data, and broader evaluation will be key to effective utilization.

References

Mohammad Gheshlaghi Azar, Mark Rowland, Bilal Piot, Daniel Guo, Daniele Calandriello, Michal Valko, and Rémi Munos. 2023. A general theoretical paradigm to understand learning from human preferences.

Sayak Ray Chowdhury, Anush Kini, and Nagarajan Natarajan. 2024. Provably robust dpo: Aligning language models with noisy feedback.

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*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Llm.int8(): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*.

Xueguang Ma, Qian Liu, Dongfu Jiang, Ge Zhang, Zejun Ma, and Wenhu Chen. 2025. General-reasoner: Advancing llm reasoning across all domains.

Lester James V. Miranda, Yizhong Wang, Yanai Elazar, Sachin Kumar, Valentina Pyatkin, Faeze Brahman, Noah A. Smith, Hannaneh Hajishirzi, and Pradeep Dasigi. 2025. Hybrid preferences: Learning to route instances for human vs. ai feedback.

Pulkit Pattnaik, Rishabh Maheshwary, Kelechi Ogueji, Vikas Yadav, and Sathwik Tejaswi Madhusudhan. 2024. Curry-dpo: Enhancing alignment using curriculum learning ranked preferences.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model.

A Full Appendix

A.1 Direct Preference Optimization (DPO) Losses

The following describes the mathematical formulations and motivations for the three loss functions we explored for our reward model: Direct Preference Optimization (DPO), Robust DPO (rDPO), and Identity Preference Optimization (IPO). Each loss function is designed to align model outputs with human preferences, while addressing different challenges in preference-based learning.

Direct Preference Optimization (DPO) is a method for training language models to better align with human preferences by directly optimizing a policy based on pairwise preference data. The DPO objective is:

$$\mathcal{L}_{\text{DPO}} = -\mathbb{E}_{x,y_w,y_l}\left[\log \sigma\left(\beta\left(\ln \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)}\right) - \beta\left(\ln \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right)\right)\right]$$

Robust DPO (rDPO) Loss extends the DPO framework to handle noisy or mislabeled preference data. By accounting for the possibility that some preference labels are flipped, rDPO provides more robust training and better generalization, especially in real-world scenarios where data quality may be imperfect.

$$\mathcal{L}_{\text{rDPO}} = \frac{1}{1-2\varepsilon}\left[(1-\varepsilon)\,\mathcal{L}_{\text{DPO}}(y_w, y_l) - \varepsilon\,\mathcal{L}_{\text{DPO}}(y_l, y_w)\right]$$

where ε is the noise rate.

Identity Preference Optimization (IPO) Loss prevents the model from becoming overconfident by targeting a specific gap τ in the log-ratio:

$$\mathcal{L}_{\text{IPO}} = -\mathbb{E}_{x,y_w,y_l}\left[\log\left(\left(\ln \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)}\right) - \left(\ln \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right) - \frac{1}{2\tau}\right)^2\right]$$

A.2 gsm8k-gpt Prompt

Listing 1: Prompt used to generate wrong choices for GSM8K

```
chat = Chat.create(f"gsm8k_{idx}",
  instruction_prefix="Answer the question
    briefly without any further interaction")

q = (
  "I have a question with the corresponding
    answer for you. "
  "I want to do a multiple choice question. The
    right choice is "
  "the provided answer and you should generate
    the wrong choice. "
  "Can you please modify the correct answer such
     that "
  "it's the wrong choice (you can choose to
    slightly modify the "
  "reasoning but it should stay the same length)
    ? "
  "Don't mention anything about the instruction
    in your answer.\n"
  f"Question: {question}\nCorrect Answer: {
    answer}\n\nWrong answer:"
)
choices = [
  chat.ask(q).content,
  chat.ask("Can you generate another one?\nWrong
     answer:").content,
  chat.ask("And a third one?\nWrong answer:").
    content
]
```

A.3 RAG Datasets

Dataset	Type	Size
SciQ	Textbook	2,647
Larger SciQ	Textbook	10,482
Generated SciQ	Generated Data	2,647
StemQ	Generated Data	667
ai2_arc	Generated Data	500
Preference Pairs	Generated Data	476

Table 4: Datasets used for the training embedding model, data generation and RAG documents

A.4 Training Hyperparameters

Hyperparameter	MCQA model	Reward model (Instr. tuning / DPO)
Learning rate	10^{-5}	$2\times10^{-5}/10^{-6}$
Number of epochs	3	2 / 3
Batch size	8	2 / 1
Max. sequence length	302 tokens	512
Optimizer	AdamW	AdamW
FP16 training	Enabled	Enabled

Table 5: Training configuration for all models

A.5 Analysis of the `allenai/multipref` Subset

We further analyze the behavior of our DPO-R model on preference pairs originating from the `allenai/multipref` dataset within our custom test set. This subset is particularly useful for diagnostic purposes, as it includes metadata for each preference pair such as question type and subject area.

Type of question	Nb of pref pairs	Reward acc.
Closed QA	18	1.00
Generation	30	1.00
Open QA	190	0.89
Coding	110	0.88
Brainstorm	16	0.87
Chat	12	0.83

Table 6: Reward accuracy by question type.

Table 6 reports performance across the most represented question types (10 examples).It shows that our model performs best on Closed QA and Generation, which often feature binary correctness (e.g., factual correctness or clearly flawed output) or identifiable errors (e.g., hallucinations or inappropriate language). Conversely, performance drops on more subjective formats such as Chat and Brainstorm, where model preferences may depend on subtle stylistic or interpersonal cues that are harder to learn and evaluate consistently.

Subject	Nb of pref pairs	Reward acc.
Veterinary Medicine	24	1.00
Generation	30	1.00
Physics	42	0.95
Mathematics	52	0.92
Computer Science	186	0.90
Biology	24	0.83
Chemistry	12	0.83
Climate Sciences	28	0.78

Table 7: Reward accuracy by subject.

Table 7 highlights similar trends at the subject level. The model shows strongest accuracy on Veterinary Medicine, Physics, and Mathematics, and weakest performance on Biology, Chemistry, and Climate Sciences. This pattern may be attributed to differences in domain representation in the training data—technical and math-heavy content is likely overrepresented. Moreover, the criteria for alignment in mathematically structured fields (e.g., logic, correctness) may differ substantially from those in domains like climate science, where interpretability, relevance to real-world phenomena, or nuanced reasoning play a larger role.

These findings suggest that while DPO-R handles objective or clearly-scoped tasks well, it remains limited when preferences depend on complex, domain-specific judgment or less deterministic criteria.

A.6 Models HuggingFace Repositories

Model	HuggingFace Repo
4-bit quant	klusertim/MNLP_M3_quantized_model
8-bit quant	klusertim/MNLP_M3_quantized_model_8bit
4-bit quant (no double)	klusertim/MNLP_M3_quantized_model_4bit_noDouble
4-bit + QLoRA	klusertim/MNLP_M3_quantized_model_4bit_adapter

Table 8: Models with their respective HuggingFace repositories

A.7 Comparison of Document Lengths

Métrique	Short SciQ	Long SciQ
Top_k=5	0.4442	0.4364
Top_k=10	0.4364	0.4260
Top_k=15	0.4325	0.4338
Top_k=20	0.4208	0.4312
Mean	0.4330	0.4320

Table 9: Accuracy with different Top_k.

A.8 Impact of RAG Documents

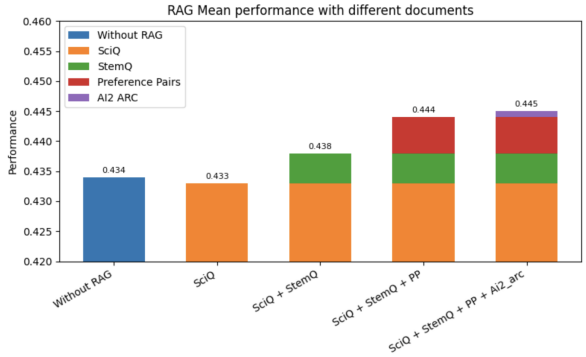

Figure 3: Mean accuracy for $top_k \in \{5, 10, 15, 20\}$.

A.9 Comparison of Textbook vs Generated Data

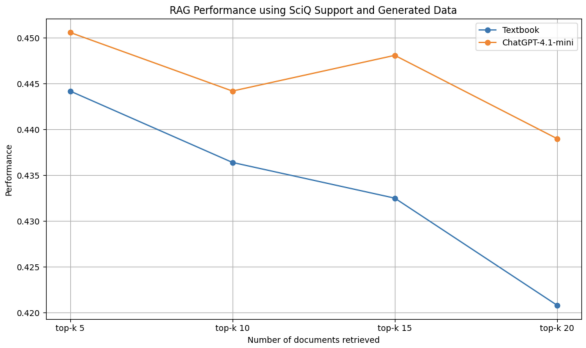

Figure 4: SciQ Textbook vs Generated support accuracy.

A.10 GPT Models Comparison

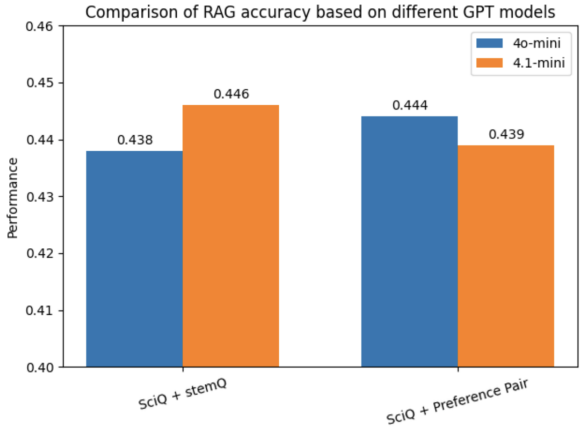

Figure 5: Mean accuracy across GPT variants.

A.11 Embedding Model Accuracy

Embedding model	Mean Accuracy
`gte-small`	0.442
`abhinand/MedEmbed-base-v0.1`	0.451
`Our embedding Model`	0.450

Table 10: Retrieval accuracy (top-k=5, cosine, full docs).

A.12 Prompt for Data Generation

Task

Write a Master-level, self-contained content that supplies all the information a reader would need to deduce the correct answer to the question below. Do not restate the question or reveal the answer explicitly.

QUESTION

`{q}`

CORRECT ANSWER

`{a}`

Writing guidelines (follow strictly):

1. Length: 200–400 words, single paragraph.
2. Style: concise, high information density, use field-specific terminology.
3. Depth: include advanced theories, formulas and explain intermediary steps if necessary.
4. Context: situate the topic within its broader scholarly framework.
5. Output: plain text only, no headings, lists, or citations of the source.