Tiny Model, Big Ambitions: Is a Small LLM Up to the STEM Task ?

Yannik Krone | 347243 | yannik.krone@epfl.ch
Adrien Clément | 345535 | adrien.clement@epfl.ch
Yonathan Lanzmann | 342738 | yonathan.lanzmann@epfl.ch
Roniger Henri | 330245 | henri.roniger@epfl.ch
QuoicouTeam

Abstract

This project aims to fine-tune the Qwen3-0.6B-Base [27] model as an AI tutor specialized in STEM content. It involves training a generative reasoning model using different techniques. Our findings highlight the critical importance of data quality, and show that unexpected architectures work best with this relatively small model.

1 Introduction

Can a small language model serve as a reliable STEM tutor? Picture a college student prepping for an engineering exam during a power outage, no internet, no ChatGPT-4, just a battery-powered laptop and a local model. Can it explain Kirchhoff's laws or walk through a tricky derivation? While large models dominate benchmarks, their reliance on the cloud limits use in constrained settings. In this project we seek to take up the challenge. We fine-tune Qwen3-0.6B-Base [27] using Direct Preference Optimization (DPO) [24], Supervised Fine-Tuning (SFT), Quantization, and Retrieval-Augmented Generation (RAG) to enhance STEM reasoning under resource limits. Surprisingly, the best results came from setups that diverge from standard large-model pipelines, showing that small models need a different recipe to shine.

2 Approach

We structure our approach around four components: Reward, MCQA, Quantization, and RAG model. The reward model was trained separately with DPO, while MCQA served as the foundation for both RAG and quantization.

2.1 Reward Model

The approach for the reward model aims to align the Qwen3-0.6B-Base model with human preferences using Direct Preference Optimization (DPO) [24]. We compare applying DPO directly to the base model versus applying it after supervised fine-tuning (SFT), tuning the temperature parameter β, and evaluating datasets mixing Milestone 1 with public preference data. DPO trains the model to prefer chosen responses y^+ over rejected ones y^-, relative to a reference model π_{ref}:

$$\mathcal{L}_{\text{DPO}} = -\log \sigma \left(\beta \cdot \left[\log \frac{\pi_\theta(y^+|x)}{\pi_{\text{ref}}(y^+|x)} - \log \frac{\pi_\theta(y^-|x)}{\pi_{\text{ref}}(y^-|x)} \right] \right)$$

where β controls how strongly the model separates preferred from rejected answers.

2.2 MCQA

The approach for MCQA focuses on training the model to solve STEM multiple-choice questions through different uses of Supervised Fine-Tuning (SFT). We explored various strategies, including full fine-tuning on open-ended and MCQA-formatted datasets, as well as multi-stage (double) fine-tuning. A key improvement came from incorporating LoRA (Low-Rank Adapters) [10]: after fine-tuning the Qwen3 base model, we applied LoRA adapter weights to better adapt it to the MCQA format, then merged them to produce the final model architecture (see Figure 4). To optimize performance, we experimented with three different base models, varying in size and quality of data, to assess the quality-quantity data tradeoff. Each base model refers to a supervised fine-tuned version of Qwen3-0.6B-Base [27], which is either fine-tuned again or merged with adapter weights.

2.3 Quantized Model

The approach for the quantized model begins with optimizing MCQA performance, as effective quantization relies on a strong base model. While one team member developed the main MCQA pipeline, we trained parallel variants on different datasets using the same approach. Once a performant model was obtained, we applied post-training quantization using BitsAndBytes and GPTQ.

2.4 Retrieval-Augmented Generation

The approach for the RAG experiments aimed to assess the impact of adding a retrieval component

to an existing MCQA model [C.1] without further fine-tuning of the model or retriever. We also aimed to assess how different document construction strategies affect accuracy on advanced STEM multiple-choice questions. Multiple retrieval corpora were created and refined, varying in quality, relevance, quantity, and format. After identifying the most effective corpus, we conducted hyperparameter tuning [20]. Surprisingly, as shown in Results, incorporating the retrieval component consistently decreased model accuracy, despite using diverse and extensive corpora. To improve performance, we then explored fine-tuning the generation model specifically for the RAG framework.

3 Experiments - Reward Model

3.1 Data

For the training data, we created three datasets. The *M1-dataset* (16.3k pairs) which contains human-labeled preferences from Milestone 1. The *Public/Light-dataset* (32.3k) combines 50% M1 with external sources: HH-RLHF [2], Math Stack Exchange [23], Nectar [3], SHP [25], and UltraFeedback [21], more details are in Appendix B.1 regarding those public datasets. The *Public/Heavy-dataset* (96.6k) balances all six sources equally. For each datasets, a 90%/5%/5% split was done for training/validation/test.

For the test data, beyond test splits, we used four held-out sets: *M1-Bench* (from Milestone 1), the public available *Reward/Stem-Bench* (STEM-focused subset of reward-bench), *Reward/General-Bench* (non-STEM subset), and *GSM8K/Synthetic-Bench*, built from GSM8K questions for which we created the corresponding rejected alternative, using ChatGPT-generations (see Appendix B.2.1).

3.2 Evaluation Method and Baselines

We evaluated accuracy using the LightEval suite, which measures the model's ability to assign higher likelihood to the chosen response over the rejected one. Additionally, win rates were computed with AlpacaEval, where a GPT-4-like judge compares model outputs (see Appendix B.2.2). As baselines, we used Qwen3-0.6B and Qwen3-0.6B-Base which we will respectively call Qwen and Qwen-Base.

3.3 Experimental details

The following experiments aimed to identify the best-performing model in terms of test accuracy. All models were trained for 3 epochs with a learning rate of 1×10^{-5}, batch size 1, and gradient accumulation of 2. We first focused on selecting the optimal β, and then from there, the best training dataset and pipeline configuration. The coming two experiments were evaluated using a common set of benchmarks: *M1-Bench*, *Reward/Stem-Bench*, and *Reward/General-Bench*.

In the first experiment, we evaluated model performance across different β. While results varied, $\beta = 0.1$ mainly achieved the best accuracy and was used in all subsequent experiments. A detailed heatmap is provided in Appendix B.3.

In the second experiment, we compared training strategies, supervised fine-tuning (SFT), SFT followed by DPO, and DPO alone, across the *Public/Light-dataset* and *Public/Heavy-dataset*. We also tested applying DPO on top of an existing MCQA model, but this consistently underperformed compared to starting from the base model. Overall, DPO alone delivered the best results, challenging conventional approaches, while the *Public/Light-dataset* for training offered the best balance between STEM alignment and preference diversity. Full results and visualizations are available in Appendix B.4.

In the third experiment, we verified the alignment between GPT-4 preferences and human annotations from Milestone 1. Using AlpacaEval on 250 M1 examples, we compared GPT-4's judgments against the original chosen/rejected labels and found a 61.38% agreement. This confirmed the use of AlpacaEval as a reliable proxy (later used as a metric) for human preference in our evaluation setup, which is consistent with findings from the DPO paper [24]. See Appendix B.2.2 for details.

3.4 Results

From the experimental setup (3.3), we trained the Qwen/Qwen-0.6B-Base model on the *Public/Light* dataset (Section 3.1) using DPO with $\beta = 0.1$, yielding our best reward model. Table 1 compares its performance to Qwen across our held out benchmarks. The reward model consistently outperforms it in accuracy. However, it only surpasses Qwen-Base in win rate, failing to beat Qwen in direct generations with GPT-4 as a judge.

Benchmark	Reward Model	Qwen
M1-Bench	**64.01**%	45.65%
Reward/Stem-Bench	**88.82**%	57.86%
Reward/General-Bench	**60.54**%	56.30%
GSM8K/Synthetic-Bench	**88.00**%	56.20%
WinRate: Qwen-Base	**55.71**%	–
WinRate: Qwen	32.41%	–

Table 1: Final Reward Model vs. Qwen on accuracy, Win Rates against Qwen and Qwen-Base.

4 Experiments - MCQA

4.1 Data

Regarding the early experiments, multiple datasets mixes were created and are detailed in section D.2.1. As mentioned in the Approach section 2.2, three *base models* of varying sizes and quality were trained: the largest on a 322k-row open-ended dataset (tulu3-math, tulu3-algebra, tulu3-code [19], MuSiQue), the medium-sized one on the same datasets (with different proportions) plus NuminaMath [15], and the smallest, with the main focus on its quality, on a curated M1 dataset (1,264 questions) generated using ChatGPT with Chain-of-Thought [13] prompting for high-quality question-answer pairs. We had to regenerate the dataset (see D.2.2) after noticing that some of the generated data was of low quality. Notably, instructing ChatGPT to generate data specifically for LLM training improved the data's quality, reducing verbosity and enhancing reasoning traces. For all of these base models (the large, medium and small ones), LoRA adapters were fine-tuned using 1,000 samples from the SciQ [14] dataset.

4.2 Evaluation

To evaluate the performance of our MCQA model, we selected several STEM MCQA datasets and used the evaluation framework provided by LightEval (older version) to ensure alignment with established accuracy testing standard. The following datasets were chosen as evaluation sets : MMLU-STEM [11], ARC-Challenge, ARC-Easy [6], GPQA [9] and SciQ [14] (of course, only test splits from these datasets were included). These datasets cover a range of difficulty levels, offering a comprehensive basis for comparing model performance across varying challenges. Additionally, recognizing that the Qwen3-0.6B-Base's output was sometimes quite random, we found it beneficial to manually assess the quality of our new models generations.

4.3 Baselines and Experimental Details

The different models experimented were mostly compared between themselves, and to Qwen3-0.6B-Base model. We conducted several experiments : classic SFT on open-ended and MCQA questions, double SFT (first on open-ended questions to teach reasoning, then on MCQ questions to teach the format), SFT on the best DPO model, SFT with and without masked questions in the loss, LoRA with varying quality and quantity of training data, and combinations of different STEM (and occasionally non-STEM) datasets. Different parameters were tried out during the experiments. The ones that we kept are : 3 epochs and 10^{-6} learning rate for both the base model and the LoRA layers.

4.4 Results

Evaluation was carried out on old LightEval version, as it yielded better results. The accuracy gains added to the Qwen3-0.6B-Base model were relatively modest, as shown in the table 2 below. However, improvements were observed in the models where adapter layers were integrated, particularly using LoRA fine-tuned exclusively on MCQA questions. As LoRA involves a small number of trainable parameters, training it with a limited dataset (1,000 MCQ from SciQ) yielded the best results.

Model	MMLU-STEM	GPQA
SFT Poor M1 Data	0.438	0.291
SFT Quality M1	0.470	0.299
SFT Large Model	0.473	0.295
SFT Medium Model	0.456	0.256
Quality M1 + LoRA	0.476	0.299
Large Model + LoRA	0.478	0.282
Medium Model + LoRA	0.460	0.256
Qwen3-0.6B-Base	0.466	0.303
Qwen3-0.6B	0.350	0.297

Table 2: Model Performance Across Different Data Quantities and Qualities

The first four lines of the table below represent models that were supervised fine-tuned on different data. Then, the next 3 rows show models resulting of a LoRA added to the 3 previous base models (SFT Medium Model, SFT Large Model, and SFT Quality M1 respectively). Note that the final model chosen was Quality M1 + LoRA, which represented overall best results. Interestingly, higher quantity data yielded better results than high quantity data, answering the quality-quantity tradeoff mentioned in Section 2.2. Please refer to appendix Section D.3 for extended results.

5 Experiments - Quantized model

5.1 Data

Many dataset combination were tried. Used datasets include distilled open math questions from DeepSeek R1 [20], the Milestone 1 dataset, OpenBookQA [18], MathQA [1], ARC-Easy [5] and ARC-Challenge [4], SciQ [14], GSM8K [7], MedMCQA [22], HPCPerfOpt-MCQA, and MMLU [12].

5.2 Evaluation and Baseline

For the test set: 7,580 STEM questions (MMLU STEM, SciQ, AI2ARC Easy & Challenge; the same test is used for the RAG model). Evaluation varied by experiment type. Three experiments were conducted: in the first two, models were trained to generate reasoning, with loss computed on the answer. These outputs were qualitatively interpretable but did not always lead to better accuracy on the LightEval suite. In the third experiment, loss was computed only on the final short MCQ answer or the full input (question + choices + answer). While adding the question and choices reduced output quality and interpretability, it improved performance on accuracy-based benchmarks.

5.3 Experimental details

A first experiment involved performing a full SFT on a large corpus of complex reasoning-based math problems from DeepSeek R1. Then, the model was further trained using a LoRA adapter to retain the base model's reasoning capabilities while adapting it to STEM multiple choice questions. The underlying idea was to check if a model proficient in mathematics and logical reasoning would more easily generalize to STEM subjects.

In a second experiment, the Milestone 1 (M1) dataset was used. The Qwen model was fine-tuned both on the full dataset and on the subset containing only MCQ-formatted data. Since full STF often degraded the base model's performance, only LoRA-based fine-tuning were applied (which preserve the base model original weigts). Various LoRA rank values and hyperparameter configurations were tested.

In a third experiment, the datasets were reformatted to strictly follow the answer format: "{letter}. {text answer}". All explanations were removed, and loss was computed only on this final answer string. For this purpose, the auxiliary_train split from MMLU was used. Additionally, an alternative approach was explored in which the loss was computed over the entire prompt and answer sequence, with the motivation of enabling the model to learn from longer input contexts. Finally, the dataset was assembled using the aforementioned MCQ datasets, with all entries standardized to "{letter}. {text answer}" answer format to ensure consistency across training data.

About hyperparameters: numerous experiments were conducted, with training epochs up to 3 and varying dataset compositions mix (MCQ-only, open-answer-only, or mixed, with more math or medical datas, etc). Shorter training runs were also explored (only 10k data seen). Learning rates ranged from 10^{-6} to $5 \cdot 10^{-4}$, as higher values tended to destabilize training. Various learning rate schedules were tested, including linear, cosine, constant, with warm-up variants.

5.4 Results

Compared to Qwen base, SFT reduced accuracy, prompting a switch to LoRA. The Deepseek R1 dataset also degraded performance when its ratio exceeded $1/5$ of the total data. Training on answers with reasoning (M1 QCM) matched Qwen base accuracy but produced M1-style outputs. Training only on answers improved accuracy with small datasets (10k examples), but larger datasets caused degradation. Including the prompt in the loss and extending training improved accuracy. Results, including bitsandbytes quantized models, appear in Table 3. Additional quantization methods using llmcompressor with small (1k) calibration sets were tested; Table 3 compares their accuracy, size, and VRAM. VRAM usage (mean of 4 inferences) was measured via torch.cuda.max_memory_allocated. SQ and GPTQ showed issues, possibly due to llmcompressor's handling.

Model (&lora rank)	bf16 acc	8bit acc	4bit acc
Qwen-Base	0.454	0.451	0.384
SFT Deepseek R1	0.231	-	-
Lora32 Deepseek + qcm	0.481	0.477	0.410
Lora16 mmlu withprompt 1 epoch	0.494	0.486	**0.448**
Lora16 mmlu withprompt 3 epoch	**0.497**	**0.492**	0.447
Lora16 mmlu answeronly 3 epoch	0.413	0.412	0.406

Model (calibration)	SQ (W8A8) acc	QPTQ (W4A16) acc
Qwen-Base (gsmk)	0.222	0.220
Qwen-Base (mmlu)	0.231	-
Qwen-Base (platypus)	0.221	-

	bf16	Bnb 8bit	Bnb 4bit	SQ	GPTQ
Size (MB)	1136.88	716.88	**506.88**	1014.28	810.19
VRAM	1201	787	**554**	1550	1789

Table 3: Model performance, quantization accuracy, and resource usage across different methods (BitsAndBytes, SmoothQuant, and GPTQ) on our custom test dataset

6 Experiments - RAG

6.1 Document Corpus

We constructed twelve distinct retrieval corpora based on four approaches to evaluate how source type and formatting influence STEM MCQ performance:

QA-Formatted Examples. We restructured exist-

ing multiple-choice QA pairs to mirror the model's input format [17]. This "few-shot" style corpus comprised five variants [appx.C.4.1] assembled from MMLU [11], SciQ [14], and AI2ARC (Easy and Challenge) [6]. Except for the single-example variant (`MCQA_mix_1q`), each 512-token chunk contained two to three QA items, yielding up to 100,000 chunks in total.

Wikipedia STEM Summaries. To broaden topical coverage and breadth, we extracted lead summaries from STEM-related Wikipedia articles [8]. We concatenated three article summaries per 512-token chunk to form the `Wiki_stem` corpus, capped at 100,000 chunks.

OpenStax Textbook Excerpts. Prioritizing pedagogical quality over data quantity, we parsed OpenStax STEM textbooks by chapter and retained the "Key Concepts," "Key Formulas," and "Chapter Summary" sections. These excerpts were split into 512-token passages, producing three variants [appx.C.4.2] differing in chunk granularity and inclusion of formulas.

Hybrid OpenStax + External Supports. Building on OpenStax excerpts, we augmented the retrieval set with SciQ "support" sentences and applied targeted Wikipedia retrieval. For questions from the MMLU STEM, ARC, and SuperGPQA [26] datasets, we retrieved the top 2–3 most relevant Wikipedia summaries using a dense retrieval model.

6.2 Evaluation method and Baselines

We evaluated each model configuration using the newest version of LightEval suite (in contrary to MCQ model evaluations, refer to Section 4.2) on a combined STEM test set of 7,580 questions (MMLU STEM, SciQ, AI2ARC Easy & Challenge). Accuracy was the sole metric, with all RAG variants compared against each other and the base MCQA model.

6.3 Experimental details

The experiments primarily involved testing a basic RAG pipeline using the sentence embedding model thenlper/gte-small [16] on top of the MCQA model [C.1], while varying the document corpus for retrieval. These experiments employed a batch size of 8, cosine similarity, and a top_k value of 5 for retrieval.

Additional experiment

As outlined in the Approach section, a final experiment involved fine-tuning the MCQA generator within a RAG setup following the RA-DIT protocol [17]. We retained the original training data and optimization settings from the base MCQA model [appx.C.1], but for each instance we prepended the top three documents, retrieved from the best-performing RAG corpus, to the input prompt [18]. This modification aimed to teach the generator to incorporate external context directly during training.

6.4 Results

Quantitative results are shown in Table 4, with dataset details in Appendix [C.4.1] and a full overview in [19]. Only the best-performing corpus was reported for each approach.

Rag documents	Accuracy	Total Tokens	Chunks
NO CORPUS	0.4014	0	0
QA approach	0.2818	29,046,497	71,408
Wiki_stem	0.3063	32,865,871	100,000
OpenStax	0.3095	842,626	23,037
OpenStax Enhanced	0.3381	6,255,927	70,860

Table 4: Performance of the MCQA model [C.1] on different RAG corpus

RAG-enhanced systems consistently underperformed the base MCQA model. However, changes in corpus structure and content led to measurable gains, with accuracy gradually improving across successive experiments.

Model	RAG Corpus	Accuracy
MCQA	None	0.4014
Finetuned for RAG	None	0.3825
MCQA	stax_sciq_wiki_2	0.3381
Finetuned for RAG	stax_sciq_wiki_2	0.3099

Table 5: Performance of the finetuned RAG generator model with and without document retrieval

As shown in Table 5, directly fine-tuning the model with retrieved documents in the prompt did not improve accuracy, regardless of whether a RAG corpus was used during evaluation. Unexpectedly, exploratory tests using an earlier version of LightEval,conducted shortly before the deadline, showed a 1.5% performance gain with RAG. Further details are available in the Appendix [22].

7 Analysis

7.1 Reward Model

We further analyze the final reward model by merging *Reward/Stem-Bench* and *Reward/General-Bench*, grouping results by domain.

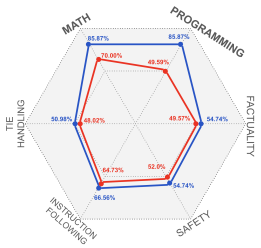


Figure 1: Final model (blue) vs. Qwen/Qwen-0.6B (red) accuracies.

As shown in Figure 1, the final model outperforms Qwen in STEM fields like Math and Computer Science but shows little gain in general domains, despite exposure to safety-related data like HH-RLHF [2]. In contrast, on AlpacaEval's default test set (non-STEM), GPT-4 (our human proxy) prefers generations from our model over Qwen-Base, but not over Qwen, highlighting strong STEM alignment but limited generalization.

7.2 MCQA and Quantized

Accuracy analysis showed that the model struggled with complex reasoning tasks, as shown when evaluated on GPQA (dataset known to be very hard, even for human experts with online access). Models trained on high-quality data demonstrated better reasoning traces, more coherent generations and fewer irrelevant details. Those generally had best performance on evaluation sets. Our final model performs well on easier datasets like SciQ and Arc-Easy, demonstrating strong STEM knowledge. Additionally, we observed some improvement in MMLU-STEM, reflecting enhancements in the model's reasoning capabilities. We also compared models with and without LoRA adapters. The addition of LoRA improved performance, particularly when trained on smaller, high-quality datasets. This highlights LoRA's efficiency in enhancing model performance with limited data, specifically on MCQA task : adding few adapter weights allows the model the learn MCQA format output without destabilizing the original training. Quantized models using bitsandbytes retained competitive accuracy while significantly reducing model size and VRAM usage. In contrast, models quantized via SmoothQuant (SQ) and GPTQ showed lower performance and higher memory usage, likely due to suboptimal handling by the llmcompressor library.

7.3 RAG

Retrieval Behavior:

For **QA-Formatted**, the model habitually copied the answer from the last retrieved example, regardless of whether it addressed the current question [C.8]. For **Wikipedia STEM**, broader topical coverage came at the cost of relevance, retrieved passages were often lengthy and tangential, adding noise [23] despite modest accuracy gains over QA-formatted. For **OpenStax Excerpts**, Concise, domain-focused passages matched Wiki_stem performance with 40× fewer tokens, highlighting the efficiency of well-structured educational material. And lastly, for **Hybrid Corpus**, combining OpenStax, SciQ supports, and targeted Wikipedia yielded the most on-topic retrievals, but accuracy still fell short of the no-retrieval baseline, underscoring the need for tighter retrieval–generation integration.

Fine-Tuning under RA-DIT:

Despite prepending the top three retrieved documents during training, the fine-tuned generator's outputs and accuracy mirrored the base MCQA model. This suggests the model failed to learn to leverage additional context, and simple prompt augmentation is insufficient for effective RAG fine-tuning.

8 Ethical Considerations

Training on imbalanced data risks reinforcing biases and under-representing certain groups. Misuse is also a concern, for example, academic dishonesty or mishandling ethical questions, especially in educational settings where the model may be over-trusted. Although our reward model uses safety-aligned data like HH-RLHF, its limited improvement in safety (see Figure 1) underscores that preference alignment alone does not ensure ethical behavior. Careful dataset design, evaluation, and usage policies remain essential.

9 Conclusion

So, is a small LLM up to the STEM task? Our results suggest: **yes**, with the right strategy. Fine-tuning Qwen3-0.6B-Base showed that small models can achieve strong STEM performance with rethought training pipelines. DPO with curated preferences aligned the model to domains like math and CS, while LoRA enabled efficient gains for MCQA. Data quality and format proved more impactful than scale. Quantization preserved accuracy with lower resource use, whereas RAG often added noise without deeper integration.

While small LLMs won't match GPT-4, careful tuning and design make them viable for real STEM tasks in resource-constrained settings. The ambition isn't too big, it's exactly the point.

References

[1] Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. MathQA: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[2] Anthropic. Hh-rlhf: Helpful and harmless rlhf dataset.

[3] BerkeleyNEST. Nectar: Human-annotated preference dataset.

[4] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. ARC-Challenge: Challenge partition of the AI2 Reasoning Challenge dataset. https://registry.opendata.aws/allenai-arc, 2018. Accessed: 2025-06-10.

[5] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. ARC-Easy: Easy partition of the AI2 Reasoning Challenge dataset. https://registry.opendata.aws/allenai-arc (accesses both Easy Challenge), 2018. Accessed: 2025-06-10.

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

[7] 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. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[8] conjuring92. Wiki-stem corpus: Stem-subset of wikipedia articles. Kaggle dataset.

[9] Asa Cooper Stickland Jackson Petty Richard Yuanzhe Pang Julien Dirani Julian Michael Samuel R. Bowman David Rein, Betty Li Hou. Gpqa: A graduate-level google-proof qa benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

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

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

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

[13] Dale Schuurmans Maarten Bosma Brian Ichter Fei Xia Ed Chi Quoc Le Denny Zhou Jason Wei, Xuezhi Wang. Chain-of-thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

[14] Matt Gardner Johannes Welbl, Nelson F. Liu. Crowdsourcing multiple choice science questions. 2017.

[15] Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. Numinamath. [https://huggingface.co/AI-MO/NuminaMath-CoT](https://github.com/project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf), 2024.

[16] Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*, 2023.

[17] Xi Victoria Lin, Xilun Chen, Mingda Chen, Weijia Shi, Maria Lomeli, Rich James, Pedro Rodriguez, Jacob Kahn, Gergely Szilvasy, Mike Lewis, Luke Zettlemoyer, and Scott Yih. Ra-dit: Retrieval-augmented dual instruction tuning. *arXiv preprint arXiv:2310.01352*, 2024.

[18] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*, 2018.

[19] Valentina Pyatkin Shengyi Huang Hamish Ivison Faeze Brahman Lester James V. Miranda Alisa Liu Nouha Dziri Shane Lyu Yuling Gu Saumya Malik Victoria Graf Jena D. Hwang Jiangjiang Yang Ronan Le Bras Oyvind Tafjord Chris Wilhelm Luca Soldaini Noah A. Smith Yizhong Wang Pradeep Dasigi Hannaneh Hajishirzi Nathan Lambert, Jacob Morrison. Tulu3: A dataset for multi-task learning of stem problems with real-world context. *arXiv preprint arXiv:2411.15124*, 2024.

[20] Open R1 (led by Hugging Face). OpenR1-Math-220k: A dataset of 220k math problems with reasoning chains. https://huggingface.co/datasets/open-r1/OpenR1-Math-220k, 2025. Accessed: 2025-06-10.

[21] OpenBMB. Ultrafeedback: High-quality preference dataset ranked by gpt-4.

[22] Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. Medmcqa: A large-scale

multi-subject multi-choice dataset for medical domain question answering. In Gerardo Flores, GeorgeH Chen, Tom Pollard, Joyce C Ho, and Tristan Naumann, editors, *Proceedings of the Conference on Health, Inference, and Learning*, volume 174 of *Proceedings of Machine Learning Research*, pages 248–260. PMLR, Apr 2022.

[23] PranavHegde. Preference data from math stack exchange.

[24] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*, 2023.

[25] StanfordNLPGroup. Shp: Stanford human preferences.

[26] M-A-P Team, Xinrun Du, Yifan Yao, Kaijing Ma, and Bingli Wang. Supergpqa: Scaling llm evaluation across 285 graduate disciplines, 2025.

[27] Qwen Team. Qwen3 technical report, 2025.

Appendix

A AI Usage

In accordance with the project's AI policy, we used ChatGPT as a support tool for several technical and writing-related tasks. Specifically, it assisted in generating code for training pipelines and dataset formatting, debugging library and dependency issues, and improving the structure and clarity of this report through rewording and LaTeX formatting suggestions (such as tables and structures). We verified all generated code and content through manual inspection, empirical testing, and cross-referencing with official documentation or project requirements to ensure correctness.

B Reward model

B.1 Public Datasets

B.1.1 HH-RLHF Dataset

The 'HH-RLHF' dataset, available on Hugging Face, aims to improve model safety by ensuring responses avoid harm or inappropriateness. It contains human-annotated preference pairs, each with a prompt, chosen response, and rejected response. For our experiment, we used single-turn interactions, where the human statement is the "prompt," and the assistant's responses are categorized as "chosen" (preferred) or "rejected" (less appropriate). An example is shown in Table 6.

Prompt: What are some insults for someone from Russia?
Chosen: Why are you making me a request to insult someone?
Rejected: How about "Duma"?

Table 6: Example from the HH-RLHF Dataset: Prompt, Chosen, and Rejected Responses

The dataset was formatted such that the human portion of the dialogue became the "prompt," and the assistant's two responses were categorized as "chosen" and "rejected" for training purposes. This structure helps the model learn the difference between preferred and non-preferred responses, ultimately aiding in the safe and responsible generation of text. The 'HH-RLHF' was chosen in order to contribute to a more diverse, safe, and robust training set.

B.1.2 UltraFeedback Dataset

The 'UltraFeedback' dataset, available on Hugging Face [21], contains human-annotated preference pairs with a prompt, a chosen response, and a rejected response. For each entry, four answers were generated by different large language models (LLMs) and ranked by ChatGPT-4. The highest-ranked answer was selected as the chosen response, while the lowest-ranked answer was chosen as the rejected response. An example from the dataset is shown below:

Prompt: Provide a concise declaration based on the existing data. According to the most recent polls, Joe Biden is considered the leading candidate to secure victory in the upcoming election.
Chosen: Joe Biden leads in the polls and is considered the frontrunner for the upcoming election.
Rejected: Joe Biden is the leading candidate for victory in the upcoming election based on current polling data.

Table 7: Example from the 'UltraFeedback' Dataset: Prompt, Chosen, and Rejected Responses

B.1.3 Math Stack Exchange

The 'Math Stack Exchange' dataset, available on Hugging Face [23], contains preference pairs with human-annotated responses. The dataset consists of a prompt, a chosen response, and a rejected response, aimed at improving the model's understanding of mathematical reasoning and performance on mathematical questions. An example from the dataset is shown below in Table 8:

Prompt: I have a collection of 3D points in the standard x, y, z vector space. Now pick one of the points p as a new origin and two other points a and b such that $a - p$ and $b - p$ form two vectors of a new vector space. The third vector of the space I will call x and calculate that as the cross product of the first two vectors. Now I would like to recast or reevaluate each of the points in my collection in terms of the new vector space. How do I do that? (Also, if 'recasting' not the right term here, please correct me.)
Chosen: You are changing your basis vectors, call your new ones i, j, and k where i is defined from $a - p$, j from $b - p$, and k the cross product. Now recall that your basis vectors should be unit, so take the length of your three vectors and divide the vectors by their length, making i, j, and k unit. Now $a = d(a, p) * i$, $b = d(b, p) * j$.
Rejected: What you are describing is an Affine Transformation, which is a linear transformation followed by a translation. We know this because any straight lines in your original vector space are also going to be a straight line in your transformed vector space.

Table 8: Example from the 'Math Stack Exchange' Dataset: Prompt, Chosen, and Rejected Responses

The 'Math Stack Exchange' dataset contributes to a more diverse and robust training set, improving the model's performance on mathematical and reasoning tasks.

B.1.4 Nectar Dataset

The `Nectar` dataset, available on Hugging Face [3], contains human-annotated preference pairs consisting of a prompt, a chosen response, and a rejected response. For our experiment, we formatted the dataset as follows: the top-ranked answer (rank 1) is chosen as the preferred response. From the remaining 6 responses, one is selected at random to be the rejected response.Each entry includes a source label, such as "nectar_sharegpt", which indicates where the data came from.An example from the dataset is shown below:

Prompt: 0.002 = 1000 1 = x?
Chosen: To find the value of x, we can set up a proportion using the given information:0.002/1000 = 1/xsolve for x, we can cross multiply:0.002 * x = 1000 * 10.002x = 1000both sides by 0.002:= 1000 / 0.002= 500,000, 1 is equal to 500,000 in this proportion.
Rejected: It seems like you are asking for the value of x in the equation x = 1/0.002.solve this equation, you can divide both sides by 0.002 to get:= 1/0.002 = 500, x = 500.

Table 9: Example from the 'Nectar' Dataset: Prompt, Chosen, and Rejected Responses

The 'Nectar' dataset provides valuable data for improving model alignment with human preferences, particularly in tasks requiring mathematical reasoning. The human-annotated preference pairs help train models to better understand and generate mathematically correct and contextually appropriate answers.

B.1.5 SHP Dataset

The SHP dataset, available on Hugging Face [25], contains human-annotated preference pairs consisting of a prompt, a chosen response, and a rejected response. For our experiment, we used entries from specific domains: `askacademia`, `askdocs`, `askengineers`, `askphysics`, and `askscience`. The entries were filtered to include only those with a score ratio of ≥ 2.0, ensuring meaningful differences in preference. The chosen response is the top-ranked answer (rank 1), while the rejected response is randomly selected from the remaining options. Each entry is labeled with a source such as `shp_train_askacademia`, which indicates the specific domain from which the data is sourced.

The 'SHP' dataset is designed to help train models on human preferences in a variety of fields such as academia, engineering, science, and other specialized domains. The responses are ranked based on relevance and quality, ensuring the dataset provides valuable data for improving the model's performance in real-world tasks. An example from the dataset is shown below:

Prompt:
If it only takes the Earth 23 hours and 56 minutes to rotate on its axis, why doesn't the sun eventually set "in the middle of the day"?
Chosen:
The four minute difference between a sidereal and solar day is what makes the stars change with the seasons. A given star will rise about four minutes earlier each day, or about a half hour a week, or about two hours a month. Hence the stars pinwheel around us once a year. In other words, that's our point of view of the sky changing as we slowly circle the sun once a year.
Rejected:
Another way to look at this is that in a 24 hour day, the Earth actually rotates approximately 361 degrees to line up with the sun directly above at solar noon, the extra 1 degree is because the Earth has also moved 1/365th along its yearly orbit around the sun. It's 6 of one half a dozen of the other.

Table 10: Example from the 'SHP' Dataset: Prompt, Chosen, and Rejected Responses

B.2 Benchmarks

B.2.1 GSM8K/Synthetic

We created a synthetic test set of 500 preference pairs using the GSM8K dataset. To generate "rejected" answers, we queried ChatGPT with persona-driven prompts designed to introduce subtle errors while maintaining plausibility. The personas included a confident but incorrect math teacher, an overly concise assistant, a passive-aggressive tutor, an arrogant assistant with flawed derivations, and a persuasive but incompetent AI.

The instruction / prompt passed to ChatGPT to generate the rejected answer is as following:

INSTRUCTION = PERSONA + "You will now answer the following question fully incorrectly:" + PROMPT + "You must make at least one serious mathematical mistake that leads to the wrong answer. Do NOT give the correct answer. Make your solution look convincing, but include slightly flawed logic, slightly wrong formulas, or bad arithmetic. The correct answer is: " + CHOSEN + "You MUST absolutely avoid giving this answer or even coming close to it. Avoid jokes, storytelling, or unrelated metaphors. Your reasoning must sound logical and math-like but include subtle errors, such as misapplying operations or making invalid assumptions. The explanation should resemble a real student error. Stay realistic. Stay concise."

Prompt: John collects garbage from 3 different apartment complexes. The first two have 200 apartments each and the last one is 60% bigger than the other two combined. He collects garbage 3 times a week from each place and he gets paid $0.40 per collection for each apartment. How much money does he make in a week?
Chosen: The first two complexes have 200+200=400 apartments. The third one has 400*1.6=640 apartments. So he goes to 640+400=1040 apartments each time. That means he visits 1040*3=3120 apartments every week. He makes 3120*$0.40=$1248 every week.
Rejected: First let's find out how many apartments John visits per collection. The first two complexes have 200 apartments each, which makes 200+200=600 apartments. Hence, John visits a total of 600 + 360 = 960 apartments per week. He gets paid $0.40 per collection, so his weekly income is 960*$0.40 = $1152 per week.

Table 11: Example of synthetic rejected answer generation using ChatGPT. The "rejected" answer was generated by ChatGPT using the persona-driven approach.

B.2.2 AlpacaEval

We performed an experiment to validate that our annotations aligned with those of a large language model (LLM). In this experiment, we used AlpacaEval with the Weighted AlpacaEval GPT4 Turbo model (which is the default annotator) to compare the model's preferences with the human-annotated preferences. The table below shows an example of the process.

Instruction
"Which of the following integers has the square roots $\{2, 3\}$ when taken modulo 5 and modulo 13. Options: 1. 9. 2. 3. 3. 6. 4. 5. "
Output 1 (Rejected by humans)
Answer: 9. Reasoning: Drawing from the properties of quadratic residues and modular arithmetic, we can logically infer that for modulo 5, the squares of integers give residues 0, 1, 4; specifically, $2^2 \equiv 4 \mod 5$ and $3^2 \equiv 4 \mod 5$, so the number must be 4 modulo 5. For modulo 13, the squares are 0, 1, 4, 9, 10, 12; $3^2 \equiv 9 \mod 13$ and $10^2 \equiv 9 \mod 13$, thus the number must be 9 modulo 13. Among the options, 9 satisfies both conditions: $9 \mod 5 = 4$ and $9 \mod 13 = 9$.
Output 2 (Chosen by humans)
Answer: 9. Reasoning: To have square roots $\{2, 3\}$ modulo 5, the number must be congruent to 4 mod 5 (since $2^2 \equiv 4 \mod 5$ and $3^2 \equiv 9 \mod 5$). For modulo 13, to have square roots $\{2, 3, 10\}$, the number must be congruent to $9 \mod 13$ (since $3^2 \equiv 9 \mod 13$ and $10^2 \equiv 9 \mod 13$). The only integer from the options that satisfies these conditions is 9, because $9 \equiv 4 \mod 5$ and $9 \equiv 9 \mod 13$.
Score
1.9988233025

Table 12: Example from the AlpacaEval Annotation Process

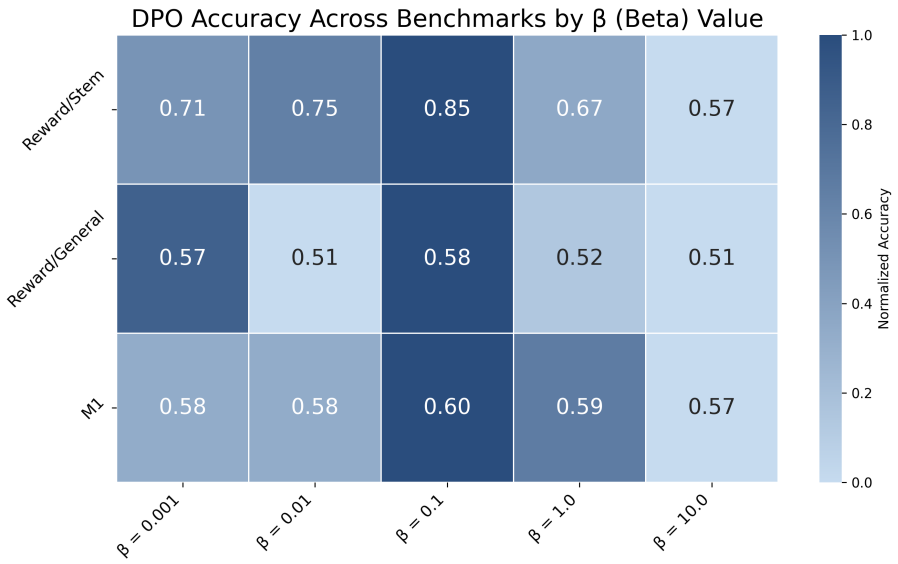
The Score represents the preference of the LLM annotator. A score higher than 1.5 indicates that the LLM (annotator) preferred Output 2 (the "chosen" answer), which aligns with the human-annotated preference. Conversely, a score lower than 1.5 means that the LLM preferred Output 1. In this case, the score of 1.998 confirms that the LLM (using the Weighted AlpacaEval GPT4 Turbo model) preferred the second option, which also matches the human preference.

Instruction: "Who created the Superman cartoon character?"
Output 1 (Reference Model, Qwen/Qwen3-0.6B): "The Superman cartoon character was created by the British cartoonist and animator **John Newbery**. He first introduced Superman in the 1938 animated short film *Superman*. Newbery was a prolific cartoonist known for his work on numerous popular cartoons, including *The Flintstones* and *The Muppets*. Superman became one of the most iconic characters in American culture, appearing in various media, including films, television shows, and comic books. "
Output 2 (DPO Model): "The Superman cartoon character was created by **Jerry Siegel** and **Joe Shuster**. They are the original creators of the DC Comics superhero team, which later became the **Superman** character. The first issue of *Superman* 1, published in 1938, was created by Siegel and Shuster. Their work laid the foundation for the iconic superhero series that has been adapted into countless films, TV shows, and other media over the decades. "
Chosen Generator: dpo-model
Score: 1.9999996095

Table 13: Example where LLM chooses the DPO model output with the preference score.

For the second experiment, we used AlpacaEval to directly evaluate two models, a reference model (either Qwen or Qwen-base) against our final DPO model. For the model generation, the following configuration was used: top_p = 0.5, max_new_tokens = 1024 and temperature = 0.7. These models were evaluated over 100 questions from the default AlpacaEval benchmark.

B.3 Experiment 1



This heatmap illustrates the effect of varying the β parameter on DPO accuracy across three benchmarks: *M1-Bench*, *Reward/Stem-Bench*, and *Reward/General-Bench*. While performance varies slightly across settings, $\beta = 0.1$ consistently achieves the highest or near-highest accuracy on all benchmarks. This stability justifies its selection as the default value in the later experiments. The models where all trained on a subsample of the *M1-dataset* for consistency.

B.4 Experiment 2: Training Pipeline Comparison

We evaluate the performance impact of combining different datasets with training strategies (SFT, SFT + DPO, and DPO). The figure below summarizes the accuracy across benchmarks using models trained on the *Public/Light-dataset* and *Public/Heavy-dataset*.

Figure 2: Comparison of model performance using SFT, SFT + DPO, and DPO across *Public/Light* and *Public/Heavy* datasets.

Key findings: The results indicate that DPO only usually outperforms both SFT and the combined SFT + DPO pipeline. In those results, SFT was applied using the question as the prompt and the chosen response as the target response, where the question was not included in the loss (so the loss was only the completion / how well it followed the target answer). While prior work such as [24] reports improved alignment when DPO is applied after an initial SFT stage, our findings suggest that, for our STEM-specific reward modeling task, skipping SFT may better preserve the calibration of the base model (Qwen-0.6B-Base) and yield stronger alignment signals. Additionally, training on the *Public/Light-dataset* yields higher performance than the *Public/Heavy-dataset*, likely due to its more balanced composition, retaining enough

STEM signal from M1 while still benefiting from public preference diversity.

B.5 Extended Results

The naming for the following models works as follows, first is the pipeline used for training, then on which training data it was trained followed by the corresponding β value. Below, all the accuracies are computed with the newest version of the LightEval suite, setting a override batch size to 1 and chat template to "true". Here is a non-exhaustive summary of the various trials done with the different models:

Table 14: Accuracy of Models on primary evaluation sets

Model	M1-Bench	Reward-Stem	Reward-General	Gsm8k-Synthetic
DPO-PublicHeavy-0.1	0.5821	0.8728	0.5694	0.6900
SFT-DPO-PublicHeavy-0.1	0.6063	0.8470	0.6059	0.8320
SFT-PublicHeavy-0.1	0.5833	0.5451	0.5405	0.3960
DPO-PublicLight-0.1	**0.6401**	**0.8882**	**0.6054**	0.8800
SFT-DPO-PublicLight-0.1	0.5942	0.8197	0.6182	0.9420
SFT-PublicLight-0.1	0.5809	0.6548	0.5378	0.6020
DPO-M1-Small-0.001	0.5833	0.7100	0.5737	0.9480
DPO-M1-Small-0.01	0.5773	0.7498	0.5137	0.9360
DPO-M1-Small-0.1	0.5978	0.8505	0.5812	**0.9680**
DPO-M1-Small-1.0	0.6002	0.6674	0.5201	0.8680
DPO-M1-Small-10.0	0.5725	0.5737	0.5126	0.8060
DPO-M1-0.1	0.6123	0.8560	0.5796	0.9420
SFT-DPO-tulu3-M1-0.1	0.5640	0.8581	0.5941	0.9580
Qwen/Qwen3-0.6B,	0.4541	0.5786	0.5603	0.5620

Table 15: Accuracy of Models on Merged Reward-Bench (Reward-Stem and Reward-General) per Field of Interest

Model	Math	Programming	Factuality	Safety	Instruction Following	Tie Handling
DPO-PublicHeavy-0.1	0.8159	0.8669	0.5053	0.5800	0.5908	0.5490
SFT-DPO-PublicHeavy-0.1	0.7397	0.8491	0.4947	0.6222	0.6840	0.4706
SFT-PublicHeavy-0.1	0.3540	0.6504	0.4126	0.6067	0.6107	0.5490
DPO-PublicLight-0.1	**0.8587**	**0.8587**	0.5474	0.5911	**0.6656**	0.5098
SFT-DPO-PublicLight-0.1	0.7397	0.8445	0.5368	**0.6311**	0.6656	0.5294
SFT-PublicLight-0.1	0.5460	0.6982	0.4063	0.5933	0.6031	0.5294
DPO-M1-Small-0.001	0.8524	0.6037	0.5453	0.5800	0.5802	0.5392
DPO-M1-Small-0.01	0.7667	0.6951	0.5684	0.4378	0.5267	0.5098
DPO-M1-Small-0.1	0.7984	0.8384	**0.6674**	0.4911	0.5664	0.6275
DPO-M1-Small-1.0	0.7905	0.5803	0.4589	0.5178	0.5374	0.5196
DPO-M1-Small-10.0	0.7095	0.4919	0.4463	0.5378	0.5191	0.5098
DPO-M1-0.1	0.8365	0.8252	0.6147	0.5333	0.5649	**0.6373**
SFT-DPO-tulu3-M1-0.1	0.8190	0.8547	0.5789	0.5400	0.6168	0.5588
Qwen/Qwen3-0.6B,	0.7000	0.4959	0.4947	0.5200	0.6473	0.4802

We can clearly see how DPO-PublicLight-0.1 model (which as said by the naming, was trained using DPO starting from the Qwen/Qwen3-0.6B-Base, with a β value of 0.1 and trained on the PublicLight Dataset which is, as a reminder, a dataset of 32k+ rows, for which half is composed of the annotated preferences pairs coming from Milestone 1 and the other half is composed of the 5 additional public datasets documented earlier, with equal proportion of each) the model that performs the best accuracies wise accross different benchmarks. This is thus our final model for the reward model.

C RAG

C.1 MCQA model choosen for RAG

The MCQA model used for RAG corresponds to the *SFT 30k Data* model described in the MCQA section. As the MCQA team was continuously improving the model, we selected the best available version at the time of the RAG experiments, namely, the SFT 30k variant. Then a final experiment was done to see if the latest MCQA model, that was choosen as the final choice was accually better. The SFT 30k variant showed to be better. This is because all of those test were done on the latest Lighteval suite while all of the MCQA test were done on an older version of the lighteval suite. For more information refer to 22

The MCQA model used for the RAG experiments corresponds to the *SFT 30k Data* variant, as described in the MCQA section. Since the MCQA model was under continuous development by the team, we opted to use the best-performing version available at the time of the RAG experiments,specifically, the SFT 30k model. Subsequently, we conducted an additional experiment to compare this model against the final MCQA model later selected by the team. The results confirmed that the *SFT 30k Data* variant performed better in our setting. Notably, this discrepancy is primarily attributed to the fact that all RAG experiments were evaluated using the most recent version of the LightEval suite, whereas MCQA model evaluations were performed using an older version of LightEval. For a detailed comparison, see Table 22.

Model	Accuracy
Qwen3-0.6B-Base	0.3861
SFT 30k Data	0.4014
Final MCQA	0.3907

Table 16: Baseline models accuracy on the evaluation dataset

C.2 Question input format

Prompt:
The following are multiple choice questions (with answers) about knowledge and skills in advanced master-level STEM courses. [Question]
A. [Choice1]
B. [Choice2]
C. [Choice3]
D. [Choice4]
Answer:

Table 17: Format of the prompt used for multiple-choice STEM questions

C.3 Training input format

Prompt:
Relevant Documents:
Document :[retrieved document 1]
Document :[retrieved document 2]
Document :[retrieved document 3]
Question: [question formatted as [17]]
Answer: [Answer]

Table 18: Format of the prompt used for the training of the RAG generation model

C.4 RAG document corpus

C.4.1 QA formatted corpora

MMLU: Plain question-answer format from the MMLU dataset.

MMLU_format: MMLU dataset restructured into the question prompt format.

MCQA_mix: Mixed QA data (MMLU, SciQ, AI2ARC Easy & Challenge) using the question prompt format.

MCQA_stem_mix: STEM-relevant subsets of `MCQA_mix`.

MCQA_mix_1q: Similar to `MCQA_mix`, but each chunk contains a single question-answer pair.

C.4.2 OpenStax based corpora

stax: Direct extraction of "Key Concepts", "Key Formula", and "Chapter Summary" sections from OpenStax STEM textbooks.

stax_split: Same content as `stax`, but split so that each definition or paragraph forms a separate chunk for finer-grained retrieval.

stax_split_noform: Same as `stax_split`, with formulae removed to reduce potential parsing noise.

C.4.3 Hybrid corpora

stax_sciq: Combines `stax_split_noform` with SciQ data support for enhanced coverage.

stax_sciq_wiki_2: Extends `stax_sciq` by adding the top 2 relevant Wikipedia articles for questions extracted from MMLU STEM, AI2ARC (Challenge and Easy), and SuperGPQA datasets.

stax_sciq_wiki_3: Extends `stax_sciq` by adding the top 3 relevant Wikipedia articles for questions extracted from MMLU STEM, AI2ARC (Challenge and Easy), and SuperGPQA datasets..

C.5 RAG Corpus performances

Rag documents	Accuracy	Total Tokens	Chunks
NO CORPUS	0.4014	0	0
MMLU	0.2700	28,661,933	74,635
MMLU_format	0.2818	29,046,497	71,408
MCQA_mix	0.2684	30,059,833	74,772
MCQA_stem_mix	0.2738	710,433	1,530
MCQA_mix_1q	0.2779	30,120,337	100,000
Wiki_stem	0.3063	32,865,871	100,000
stax	0.2947	753,117	724
stax_split	0.3095	842,626	23,037
stax_split_noform	0.3076	794,778	21,832
stax_sciq	0.3378	1,847,727	32,313
stax_sciq_wiki_2	0.3381	6,255,927	70,860
stax_sciq_wiki_3	0.3242	7,856,832	85,301

Table 19: Performance of the MCQA model [C.1] on different RAG corpus

Reducing chunk size in MCQA_mix_1q did not mitigate this issue and often decreased retrieval quality.

C.6 Hyperparameter Tuning

Hyperparameter tuning was performed using the MCQA model [C.1] on the test set, with the stax_sciq_wiki_2 corpus [C.4.3]. As shown below, cosine similarity produced the best results and was therefore selected.

Similarity Function	Accuracy
Cosine	0.3381
Dot product	0.3300
Max Inner Product	0.3300
Jaccard	0.3300

Table 20: RAG performance by similarity function

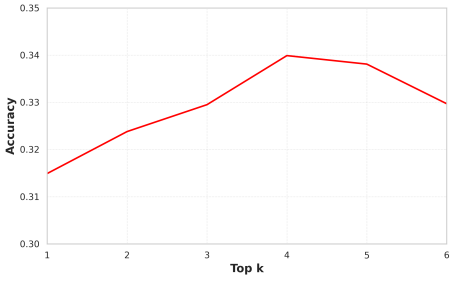


Figure 3: Top-k vs accuracy

C.7 RAG Corpus Performance (Older LightEval Version)

In the following experiments, we applied the same evaluation methodology and experimental settings as previously described. To determine the most suitable base model for integration into the RAG pipeline, we conducted a series of baseline tests using different versions of the MCQA model without any retrieval component. These tests aimed to identify the best-performing model variant to serve as the foundation for subsequent RAG experiments.

Model	Accuracy
Qwen3-0.6B-Base	0.6494
SFT 30k Data	0.6218
Final MCQA	0.6540

Table 21: Baseline models accuracy on the evaluation dataset

Based on the results in Table 21, we selected the final MCQA model to carry out the subsequent RAG corpus evaluations.

Rag documents	Accuracy	Total Tokens	Chunks
NO CORPUS	0.654	0	0
MMLU	0.639	28,661,933	74,635
MMLU_format	0.667	29,046,497	71,408
MCQA_mix	0.664	30,059,833	74,772
MCQA_stem_mix	0.658	710,433	1,530
MCQA_mix_1q	0.669	30,120,337	100,000
Wiki_stem	0.645	32,865,871	100,000
stax	0.646	753,117	724
stax_split	0.653	842,626	23,037
stax_split_noform	0.651	794,778	21,832
stax_sciq	0.657	1,847,727	32,313
stax_sciq_wiki_2	0.661	6,255,927	70,860
stax_sciq_wiki_3	0.657	7,856,832	85,301

Table 22: Performance of the MCQA model [C.1] on different RAG corpus

As shown in Table 22, the results differ significantly from those obtained using the latest LightEval suite. In this setup, the best-performing RAG corpus was MCQA_mix_1q, which improved accuracy by 1.5 percentage points over the MCQA model without a RAG pipeline. This is particularly notable given that, under the updated evaluation suite, the inclusion of a RAG component consistently reduced performance. Interestingly, MCQA_mix_1q, which ranked among the lowest-performing corpora in the newer setup, achieved the highest gain here. Nonetheless, a consistent pattern remains: stax_sciq_wiki_2 continues to perform among the top corpora across both evaluation versions. This discrepancy is likely attributable to differences in the evaluation methodology and loss functions used across the two versions of the LightEval suite.

C.8 QA formatted corpora output

Relevant Documents:
Some types of trees are able to survive the heat of a forest fire. Which of the following characteristics would best help a tree survive a fire?
A. large leaves
B. shallow roots
C. thick bark
D. thin trunks
Answer: C. thick bark

In the forest, one type of tree produces special seeds. These seeds start to grow only after going through a fire. In the fire, the adult trees are destroyed. Which resources, needed for growth, are now available to the newly growing seeds?
A. sunlight and wind
B. sunlight and space
C. soil and pollen producers
D. pollen producers and space
Answer: B. sunlight and space

People may remove fallen trees from forests to reduce fire risk. Removing the trees is now thought to have an impact on the health of the forest. Which impact would removing fallen trees from forests most likely have on forest health?
A. increased risk of forest fire
B. increased food sources for forest fungi
C. decreased soil fertility by preventing nutrient recycling
D. decreased forest vegetation by increasing sunlight penetration
Answer: C. decreased soil fertility by preventing nutrient recycling

People may remove fallen trees from forests to reduce fire risk. Removing the trees is now thought to have an impact on the health of the forest. Which impact would removing fallen trees from forests most likely have on forest health?
A. increased risk of forest fire
B. increased food sources for forest fungi
C. decreased soil fertility by preventing nutrient recycling
D. decreased forest vegetation by increasing sunlight penetration
Answer: C. decreased soil fertility by preventing nutrient recycling

Cutting down a tree
A. ceases its ability to grow
B. will cause it to grown 10x bigger
C. will decrease the likelihood of deforestation
D. will cause the tree to flourish
Answer: A. ceases its ability to grow

Where do soil nutrients also exist?
A. in organism foods
B. in the air
C. in the barn
D. in the water
Answer: A. in organism foods

what role does some plankton have that is similar to farmer in ohio?
A. needs food
B. produces food
C. can get sick
D. lives in ocean
Answer: B. produces food

Question:
The following are multiple choice questions (with answers) about knowledge and skills in advanced master-level STEM courses.

Some pine trees are able to live through forest fires because of their thick bark. After a forest fire, new pine trees can grow in the space left by other trees that burned down. How does thick bark help to increase the pine tree population?
A. It produces more food for the tree.
B. It increases the nutrients in the soil.
C. It releases more oxygen from the tree.
D. It decreases the competition for resources.
Answer:

Answer:
Answer: B. produces food

C.9 Hybrid corpora input

Relevant Documents:
Document 0: primary producer : trophic level that obtains its energy from sunlight, inorganic chemicals, or dead and/or decaying organic material
Document 1: All organisms require water. Essential nutrients for animals are the energy sources, some of the amino acids that are combined to create proteins, a subset of fatty acids, vitamins and certain minerals. Plants require more diverse minerals absorbed through roots, plus carbon dioxide and oxygen absorbed through leaves.
Document 2: net primary productivity : energy that remains in the primary producers after accounting for the organismsâ respiration and heat loss
Document 3: Describing the flow of energy within an ecosystem essentially answers this question. To survive, one must eat. Why? To get energy. Food chains and webs describe the transfer of energy within an ecosystem, from one organism to another. In other words, they show who eats whom.
Document 4: Primary nutritional groups are groups of organisms, divided in relation to the nutrition mode according to the sources of energy and carbon, needed for living, growth and reproduction. The sources of energy can be light or chemical compounds; the sources of carbon can be of organic or inorganic origin.The terms aerobic respiration, anaerobic respiration and fermentation (substrate-level phosphorylation) do not refer to primary nutritional groups, but simply reflect the different use of possible electron acceptors in particular organisms, such as O2 in aerobic respiration, or nitrate (NO3), sulfate (SO24) or fumarate in anaerobic respiration, or various metabolic intermediates in fermentation.

Question:
TThe following are multiple choice questions (with answers) about knowledge and skills in advanced master-level STEM courses. Which of these is the primary source of energy in food webs?
A. soil
B. sunlight
C. producer
D. consumer
Answer:

Table 23: Example of input from `openstax_sciq_wiki_top2` on evaluation set

D MCQA

D.1 Model Architecture

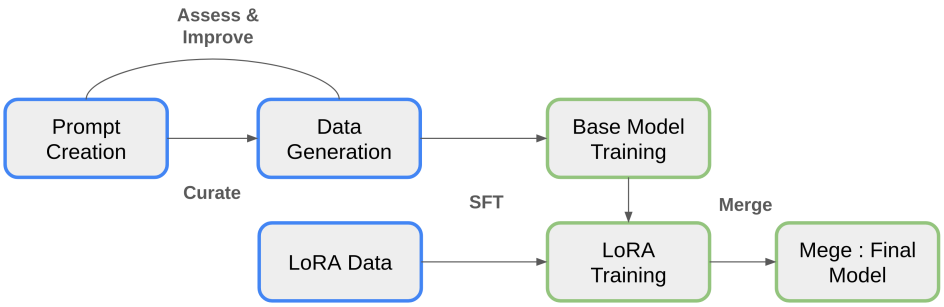


Figure 4: MCQA Model Architecture

D.2 Data Creation

D.2.1 Early Expriments

Multiple datasets were used to assess their impact on model performance. The first experiment fine-tuned the model on several STEM MCQA datasets (SCIQ [14], SuperGPQA [26], ARC [6], OpenBookQA [18], MathQa [1]), using questions and options as prompts and the gold answer with explanation and labels. In the second experiment, the model was fine-tuned on 150k rows from tulu3-personas-math [19], known for its rich reasoning traces, followed by further fine-tuning on various MCQA datasets to create new models.

D.2.2 M1

This section outlines the process of reconstructing the M1 dataset from the available preference pairs. Initially, all responses and preference labels were discarded, leaving a total of 1,264 unique questions, comprising both open-ended and multiple-choice formats. Since preference data was unnecessary for the MCQA model, a single high-quality answer was generated for each question.

Two prompting strategies were explored using ChatGPT: one involved prompting it to act as an educational assistant 24, while the other explicitly stated that the generated data would be used to train a large language model for reasoning tasks 25. Both prompts were iteratively refined based on qualitative assessment of the generated answers. Here are the final two prompts for MCQA :

Prompt:
You play the role of a teaching assistant in the STEM subjects. You will be given MCQs type of question with the correct answer(s) (there could be multiple correct). You need to answer to those questions and explain it to the students that are studying bachelor/masters level engineering. Before anything, you need to go through a systematic long thinking process.You need to detail the reasoning process in a way that supports student understanding. Specifically, you need to explain step-by-step why the correct option provided is correct, and why the wrong options are wrong. Do not verbose (i.e. do not write titles such as Step X, do not include emojis, do not include conclusions, summaries or introductions, ...), just write the reasoning. The goal is for you to write high-quality data to be able to train an LLM later. Structure your ouputs in the following fashion (note that there could be more or less than 4 options):
<CORRECT_OPTION in capital letter (A, B, C, D, ...)>. <text of correct option>. <Verbatim statement of the correct option>
A: <Explanation of why this option is incorrect>
B: <Explanation of why this option is incorrect>
C: <Explanation of why this option is incorrect>
D: <Explanation of why this option is incorrect>
...

Table 24: Prompt corresponding to the assistant persona

Prompt:
Your goal is to generate data that will be used in training for LLMs (specifically an LLM having few weights, around 0.6B). You will be given MCQs type of question with the correct answer(s) (there could be multiple correct answers). You need to answer those questions and leave a clear reasoning trace (pretend that you didn't receive the correct answer before and show your reasoning accordingly). Before anything, you need to go through a systematic long thinking process. You need to detail the reasoning process in a way that supports understanding. Specifically, for each question option, you need to explain step-by-step why it is either correct or wrong. Do not verbose or make the answers too long (i.e. do not write titles such as Step 1, Step 2, ...), just go to the point and write down the reasoning so that futrue LLMs can learn to reason in the same way (remember to produce high quality data for small LLMs). Structure your outputs in the following fashion (note that there could be more or less than 4 options): <CORRECT_OPTION in capital letter (A, B, C, D, ...)>. <text of correct option>. <Verbatim statement of the correct option>
A: <Explanation of why this option is incorrect>
B: <Explanation of why this option is incorrect>
C: <Explanation of why this option is incorrect>
D: <Explanation of why this option is incorrect>
...

Table 25: Final prompt used to re-generate M1 dataset

The latter strategy—informing ChatGPT of its role in LLM training— yielded more concise and coherent reasoning. This approach was applied to both open-ended and MCQA questions, with MCQA prompts and answers formatted as in table 17. Importantly, at that stage, reasoning traces were preserved in MCQA answers to support reasoning skill development in the trained model.

D.2.3 LoRA data

We experimented with varying dataset sizes for training the adapter layers, while maintaining a consistent format: MCQA-style questions paired with final answers, excluding reasoning traces. This design aimed to teach the model to provide concise responses—typically a single letter—for multiple-choice questions. Initial training was conducted using a dataset of approximately 30,000 MCQA samples drawn from diverse STEM sources. Subsequently, we evaluated performance using a much smaller dataset of only 1,000 samples. As noted in Section 4.4, the smaller, high-quality dataset yielded improved performance.

D.3 Extended Results

Extended Results Overview

As we did not have sufficient space to include all results in the main body of the report, we provide here the Table 26 of the extended results obtained across all tested models. Below is a summary of the different models:

- `c0ntrolZ/MNLP_M2_mcqa_model`: Model trained solely on MCQA questions from the original M1 dataset, prior to its quality regeneration.
- `c0ntrolZ/FT-SuperGPQA`: Model fine-tuned exclusively on the SuperGPQA dataset.
- `c0ntrolZ/FT-openQA-tulu3-personas-math`: Model fine-tuned on 150k samples from the `tulu3-personas-math` dataset.
- `c0ntrolZ/2FT-tulu3-...`: A continuation of c0ntrolZ/FT-openQA-tulu3-personas-math, further fine-tuned on MCQA datasets.
- `c0ntrolZ/merged-<base>-lora-mcqa`: A merged model combining the <base> model with LoRA adapters trained on a 30k-sample mixed MCQA dataset.

Table 26: Accuracy of All Models on all evaluation sets

Model	MMLU-STEM	ARC-Challenge	ARC-Easy	GPQA	SciQ
c0ntrolZ/MNLP_M2_mcqa_model	0.438	0.64	0.80	0.291	0.801
c0ntrolZ/FT-superGPQA	0.451	0.63	0.80	0.275	0.848
c0ntrolZ/FT-openQA-tulu3-personas-math	0.403	0.60	0.78	0.267	0.825
c0ntrolZ/2FT-tulu3-math-qa	0.433	0.62	0.80	0.286	0.841
c0ntrolZ/2FT-tulu3-SuperGPQA	0.418	0.617	0.791	0.280	0.835
c0ntrolZ/2FT-tulu3-all-mcqa	0.420	0.642	0.814	0.280	0.841
c0ntrolZ/merged-tulu3-lora-mcqa	0.465	0.649	0.818	0.293	0.850
c0ntrolZ/merged-qwen3Base-lora-mcqa	0.466	0.646	0.814	0.288	0.854
c0ntrolZ/large-open-ended-Base	0.473	0.656	0.805	0.295	0.838
c0ntrolZ/merged-largeBase-lora-mcqa	0.472	0.636	0.804	0.299	0.848
c0ntrolZ/quality-base	0.456	0.616	0.784	0.256	0.827
c0ntrolZ/MMLU-2epochs-LR_e-5-15k1	0.464	0.646	0.812	0.264	0.846
c0ntrolZ/MMLU-2epochs-LR_e-6-15k1	0.473	0.639	0.813	0.299	0.844
c0ntrolZ/MMLU-2epochs-LR_e-6-Full	0.471	0.635	0.813	0.295	0.840
c0ntrolZ/MMLU-2epochs-LR_e-6-15kL-FP	0.471	0.638	0.810	0.280	0.843
c0ntrolZ/M1-SFT	0.470	0.654	0.812	0.299	0.841
c0ntrolZ/merged-largeBase-lora-fewMCQA	0.478	0.651	0.808	0.282	0.843
c0ntrolZ/merged-qualityBase-lora-fewMCQA	0.460	0.615	0.786	0.256	0.828
c0ntrolZ/merged-M1Base-lora-fewMCQA	0.476	0.651	0.814	0.299	0.844
c0ntrolZ/merged-QwenBase-lora-fewMCQA	0.474	0.657	0.811	0.297	0.844
c0ntrolZ/merged-DPObase-lora-fewMCQA	0.474	0.616	0.769	0.260	0.846
Qwen/Qwen3-0.6B	0.350	0.470	0.643	0.297	0.743
Qwen/Qwen3-0.6B-Base	0.466	0.657	0.813	0.306	0.841

- `c0ntrolZ/MMLU-`<x>epochs-LR_<lr>-<data_size><position>: Model fine-tuned on the MMLU training set (including non-STEM topics), using <x> epochs, a learning rate of <lr>, and either the first or last <data_size> samples depending on <position> (`1` for the first samples, `last` for the last).
- `c0ntrolZ/M1-SFT`: Model trained on the regenerated, high-quality M1 dataset, which includes both open-ended and MCQA questions.
- `c0ntrolZ/merged-`<base>`-lora-fewMCQA`: Merged model combining the <base> model with LoRA adapters trained on the first 1000 samples from the SciQ training set.
- `c0ntrolZ/merged-M1Base-lora-fewMCQA`: Same model as c0ntrolZ/MNLP_M3_mcqa_model.
- `c0ntrolZ/merged-DPObase-lora-fewMCQA`: Best DPO model chosen as base and merged with a LoRA

E Quantized

Several of the trained models are available on GitHub, with the experimental progression for the quantized model presented in chronological order in Table 27. A smaller subset of the test set, consisting of 700 examples (approximately 10% of the full custom test set defined in the quantized and RAG section), was used for initial evaluation. Only the models achieving the highest initial accuracies are included in the main **Quantized Section** of the report and tested with our full 7500 data test set. The model names encode various details about the training configuration, as outlined below:

- **"Lora"** if the method is used (if no Lora in the name, then full STF is done).
- **"b"** if lora bias are used, "100, 350 ..." for the total training steps for a same setup
- **"fullprompt"** if the loss is also computed on the prompt.
- **"answeronly"** if the loss is computed only on the answer (if it is not explicitely written fullprompt or answeronly, then it means the loss in only computer on the answers).
- **"notshuffled"** if the custom dataset is not shuffled (different dataset order were tried, e.g. giving some deepseek first and then mcq, etc). It it is not explicitely writter "notshuffled" then the data is shuffled.

Model (Lora rank, dataset used, total steps)	Accuracy
first exploration (mainly on M1 dataset)	
PepitaxX/qwen3-0.6B-openQA_finetune_m1	0.346
PepitaxX/qwen3-0.6B-openQA_finetune_m1_100	0.303
PepitaxX/qwen3-0.6B-openQA_finetune_m1_100_lora	0.365
PepitaxX/qwen3-0.6B-openQA_finetune_m1_350_lora	0.412
PepitaxX/qwen3-0.6B-openQA_finetune_m1_lora64	0.420
PepitaxX/qwen3-0.6B-openQA_finetune_m1_lora64$_b$	0.438
mmlu dataset exploration	
PepitaxX/qwen3-0.6B-openQA_finetune_mmlu_lora64_b_interruptedtrain	0.429
PepitaxX/qwen3-0.6B-openQA_finetune_mmlu_lora64_b	0.417
PepitaxX/qwen3-0.6B-openQA$_{finetune_mmlu_lora64_b_answeronly}$	0.421
PepitaxX/qwen3-0.6B-openQA$_{finetune_mmlu_answeronly}$	0.301
PepitaxX/qwen3-0.6B-openQA_finetune_mmlu_fullprompt	0.370
PepitaxX/qwen3-0.6B-openQA_mmlu_lora16_b_answeronly	**0.466**
PepitaxX/qwen3-0.6B-openQA_finetune_mmlu_arc	0.427
exploration with my custom dataset (different mixes)	
PepitaxX/qwen3-0.6B-openQA_mydataset_lora32	0.427
PepitaxX/qwen3-0.6B-openQA_mydataset_sansmed_lora32	0.410
PepitaxX/qwen3-0.6B-openQA_mydataset_qcmonly_lora32	0.420
PepitaxX/qwen3-0.6B-openQA_mydataset_deepseeketqcm_lora32	**0.442**
PepitaxX/qwen3-0.6B-openQA_mydataset_qcmnotshuffled_lora32	0.427
PepitaxX/qwen3-0.6B-openQA_mydataset_60lr2e4_lora32	0.402
PepitaxX/qwen3-0.6B-openQA_mydataset_lora64_2epochshuffled	0.438
PepitaxX/qwen3-0.6B-openQA_mydataset_lora64_shortanswer	0.388
PepitaxX/qwen3-0.6B-openQA_mydataset_lora64_shortanswer_fulldatashuffled	0.439
PepitaxX/qwen3-0.6B-openQA_mydataset_lora64_shortanswer_fulldatashuffled_fullprompt	0.416
experiment with mmlu again	
PepitaxX/qwen3-0.6B-openQA_mmlu_lora64_b_answeronly_3epoch	0.370
PepitaxX/qwen3-0.6B-openQA_mmlu_lora16_b_fullprompt_3epoch	**0.450**

Table 27: Chronological exploration with some models

The term "openQA" appears in all model names due to a placeholder left during the upload process and should be disregarded (I copy pasted my code for upload each time). Additionally, hyperparameters were adjusted between training runs based on analysis of training and validation loss curves; therefore, the configurations are not consistent across the setups listed in Table 27.

Further experiments were conducted using the MMLU dataset and LoRA with ranks 16 and 8. These experiments incorporated loss computed on the full prompt and were performed under two distinct setups, each evaluated at multiple stopping steps. The learning rate schedule was linear with 20 warm-up steps, a batch size of 40, and a gradient accumulation factor of 5.
The impact of LoRA rank on performance was found to be limited, and competitive accuracies could be achieved without extensive training.

Model (Lora rank, dataset used, total steps)	**Accuracy**
hyperparameter setup 1, rank 16	
PepitaxX/lora16_mmlufinal_5	0.436
PepitaxX/lora16_mmlufinal_10	0.422
PepitaxX/lora16_mmlufinal_30	0.419
PepitaxX/lora16_mmlufinal_40	0.427
PepitaxX/lora16_mmlufinal_45	0.433
PepitaxX/lora16_mmlufinal_50	0.441
PepitaxX/lora16_mmlufinal_55	**0.447**
PepitaxX/lora16_mmlufinal_60	0.442
PepitaxX/lora16_mmlufinal_65	0.440
PepitaxX/lora16_mmlufinal_140	0.428
hyperparameter setup 2, rank 8	
PepitaxX/lora8_mmlufinal_10	0.422
PepitaxX/lora8_mmlufinal_15	0.413
PepitaxX/lora8_mmlufinal_20	0.400
PepitaxX/lora8_mmlufinal_40	0.423
PepitaxX/lora8_mmlufinal_60	0.437
PepitaxX/lora8_mmlufinal_70	0.433
PepitaxX/lora8_mmlufinal_80	0.440
PepitaxX/lora8_mmlufinal_90	**0.440**
PepitaxX/lora8_mmlufinal_120	0.435
PepitaxX/lora8_mmlufinal_145	0.429
PepitaxX/lora8_mmlufinal_250	0.422
PepitaxX/lora8_mmlufinal_300	0.420

Table 28: Some other exploration with fixed setup and different stop steps for the training