GemmaEdu: Elevating Scientific Learning with Modern NLP Techniques

Salim Boussofara | 311800 | salim.boussofara@epfl.ch
Oussama Gabouj | 315342 | oussama.gabouj@epfl.ch
Yasmine Chaker | 311675 | yasmine.chaker@epfl.ch
OSY

Abstract

Large language models have become essential tools, especially for students seeking assistance with complex subjects. Despite their capabilities, these models often struggle with intricate questions in advanced scientific fields. To address this, our project develops an advanced educational assistant chatbot utilizing the quantized Gemma 7B model. By employing Direct Preference Optimization (DPO) to fine-tune responses according to user preferences and integrating Retrieval-Augmented Generation (RAG) to incorporate external knowledge, we aim to enhance the chatbot's accuracy and relevance in answering multiple-choice questions. Our findings indicate that the fine-tuned model significantly outperforms baseline models in delivering accurate responses. Although we encountered limitations, such as not fully achieving benchmark performance due to testing constraints, our work demonstrates the potential of these techniques in improving educational tools. Future efforts will focus on expanding the RAG database and refining prompt engineering to further elevate the model's performance.

1 Introduction

Large language models have rapidly advanced and become crucial for students' learning. However, they still struggle with complex questions in advanced scientific fields.

Our project develops an advanced educational chatbot based on the quantized Gemma 7B model (unsloth, 2023). This model, known for its efficiency and scalability, serves as the foundation for our chatbot. To align the generated responses with user preferences, we train a generator model that produces completions ranked as preferable using DPO. This technique allows us to finetune the model to better meet the specific needs and expectations of users. Additionally, we enhance the chatbot's performance by integrating the RAG feature, which leverages external knowledge sources to provide more relevant answers. This integration ensures that the chatbot delivers precise and informative responses, thereby improving its overall effectiveness in educational applications.

2 Related Work

This section provides an overview of existing work relevant to our project, highlighting how it builds upon and differentiates from prior efforts.

Pre-trained generative models, such as Transformers, are effective for tasks like translation, summarization, and question answering. These models use structures like encoder-decoder (e.g., BERT, T5) or decoder-only (e.g., LLaMA, Mistral, Gemma). The Gemma-7B model (Google, 2024), developed by Google DeepMind, is lightweight, scalable, and efficient. Techniques like quantization and Low-Rank Adaptation (LoRA) enhance performance and allow deployment on various platforms. Unsloth's integration of Gemma-7B combines these methods, optimizing resource usage and ensuring robust performance for our educational chatbot. Despite these advantages, finetuning is necessary to adapt the model to specific educational contexts and user needs.

Methods like Self-Supervised Fine-Tuning (SFT) and DPO are crucial for task-specific adjustments. SFT improves model performance by fine-tuning on specific datasets (Jiang et al., 2024). DPO enhances response accuracy and relevance by aligning model outputs with user preferences (Kim et al., 2023). Our project uses DPO to better address students' nuanced needs, setting it apart from generic fine-tuning approaches.

RAG enhances response quality by dynamically accessing external knowledge. RAG's effectiveness across various tasks has been demonstrated (Lewis et al., 2020), showing significant improvements in accuracy and contextual relevance. Integrating RAG ensures our chatbot provides more relevant and informative responses, which is criti-

cal for educational utility.

Prompt engineering techniques like chain-of-thought prompting and few-shot learning also improve model performance. Chain-of-thought prompting (Wei et al., 2022) helps generate logical responses by breaking down tasks into steps, while few-shot learning (Brown et al., 2020) enhances adaptability and accuracy with minimal examples. Prompt engineering is used to create preference data for training and to develop the best prompts.

By building on these advanced models, the goal of our project is to enhance educational chatbot capabilities demonstrating the benefits of integrating advanced optimization and retrieval methods.

3 Approach

The first critical decision to make is the choice of the model to be used. We needed a model that balances the number of parameters with strong capabilities in reasoning, question answering, and summarization which are the key features that should be leveraged by an educational chatbot. We selected the Unsloth/Gemma-7b-bnb-4bit model (unsloth, 2023), a quantized version of the Gemma 7B model. The Gemma 7B model (Google, 2024) is a text-to-text, decoder-only large language model (LLM) pretrained using Google's web dataset, which is known for its high performance in these tasks outperforming other models of similar size (kagglepro, 2024) such as Llama 2 (Meta, 2023) and Mistral (AI, 2024). However, with its 7 billion parameters, it demands substantial computational resources.

Quantization in the Unsloth model reduces the precision of the model parameters, decreasing the model size and increasing inference speed without a significant loss in performance. To further reduce the number of parameters, we utilize LoRA, which modifies only a small subset of the parameters, enabling efficient finetuning. Consequently, the model trains with approximately 50 million parameters using a sigmoid loss function.

3.1 Model Finetuning

Given the model's high computational requirements, we employed the Tesla V100-PCIE-32GB GPU available from the IZAR SCITAS cluster to fine-tune it and align it with our specific educational context. We implemented two main strategies for this fine-tuning process: (1) SFT followed by DPO finetuning, (2) DPO-only finetuning. The goal is to maximize both the margin (difference between the preferred and non-preferred scores) and accuracy of the model responses.

$$\mathcal{L}_{DPO}(\pi_\theta; \pi_{ref}) = -E_{(x,y_w,y_l)\sim\mathcal{D}} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{ref}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{ref}(y_l|x)}\right)\right].$$

3.2 Preference dataset generation

To train these models, a preference dataset is created by EPFL Master students in the context of the Modern NLP course. Each student has been given 100 exam questions from various EPFL courses in the following domains: computer science theory, computer software, computer systems, electrical engineering, theoretical physics, artificial intelligence, and machine learning. They were asked to use GPT API to create 2 different answers and choose the best one. To ensure a highly useful dataset, both answers should be correct and providing an acceptable explanation, thus offering a concrete choice to the annotator. We thought of various techniques, including Chain-of-Thought (CoT) prompting and few-shot learning, to enhance the model's adaptability for generating high-quality responses. In the end, we chose to standardize the prompts with the following instruction: *"You are an educational assistant. We will present various questions, and we want you to briefly and then provide an explanation of the final result."*. Then, the question is presented, and if it is a MCQ, the possible answers are listed alongside the prompt: *"Select one (or more) answer(s):"*. To generate the first answer, we add, the task T = *"Give only the number(s) of the correct answer(s)."* To generate the second answer, we instruct, *"Analyze the question carefully: Let's think step by step:"*. We then take this answer, add it to the initial prompt as a hint, and finally provide the task T.

3.3 RAG pipeline

In the pipeline illustrated in Figure 1, we employ the Recursive Character Text Splitter to divide documents into sections, with each section containing 500 characters and overlapping by 25 characters. This approach is favored over splitting the text evenly to avoid dividing information between two chunks. To create embeddings, we use the llama_index embeddings provided by Hugging Face, specifically employing the BAAI/bge-small-en-v1.5 (of Artificial Intelligence, 2023) model. This particular model is chosen due to its exceptional ability to generate high-quality embeddings for our document segments. The created embed-

dings are subsequently saved in the Chroma DB database (Team, 2023). Chroma DB is selected for its ability to effectively manage and retrieve high-dimensional embeddings, making it easier to quickly retrieve relevant chunkcs during inference.

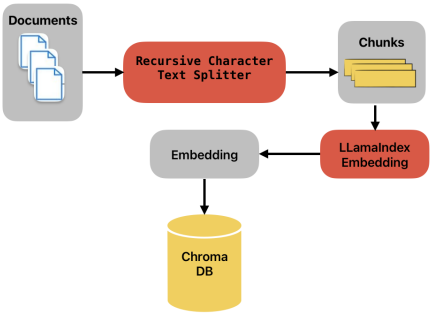

Figure 1: RAG pipeline

3.4 From Sentences to a Single Letter and RAG Integration

Our model has been trained on long sentences that contain both the answer (not always accompanied by the corresponding letter) and an explanation. The challenge is to make it select a single letter without losing the knowledge acquired during training. Using Chain of Thought (CoT), the model first generates an answer with a brief explanation. This initial prompt, concatenated with the acquired answer, forms the basis for the second prompt, where the model is constrained to respond with a single token. For effective prompting, we need two components: **Verbalizer** that matches a word to each label and a **Template** to incorporate the review with one masked token predicting one of the verbalizers. The success of this method depends on testing various templates and verbalizers to find the most efficient configuration.

Figure 2 illustrates the inference pipeline coupled with the RAG pipeline: The process starts with embedding the question using LLamaIndex Embedding, and storing it in Chroma DB. The retriever fetches the most relevant context, combined with the initial question in CoT Prompt 1. Gemma processes this to generate a CoT Response, which is then fed back into Gemma to produce the final MCQ response using the Template. Finally, we use the Verbalizer to extract the correct letter.

To assess the performance of our model, the MMLU benchmark from which only subjects relevant to STEM are kept is performed on different baselines and our model to identify the improvement that we provided.

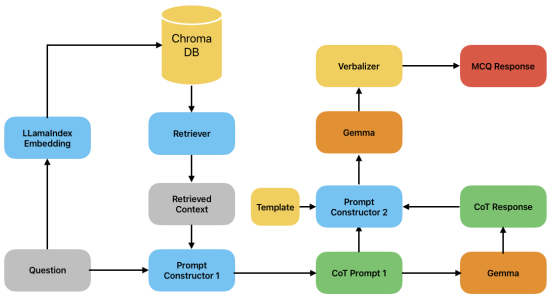

Figure 2: Inference Workflow with RAG integration

4 Experiments

4.1 Data

In this section, we discuss the datasets utilized for refining and assessing our model, along with their sources and the specific preprocessing steps applied to guarantee quality and relevance.

4.1.1 Argilla Ultrafeedback-binarized-preferences-cleaned dataset (Bartolome et al., 2023) for fine-tuning

This dataset is an enhanced version of the "Argilla/UltraFeedback-Binarized-Preferences" dataset (Cui et al., 2023), it comprises 60,917 samples consisting of instructions in various domains, half of which are scientific. While this dataset is initially intended for DPO tasks, we use it for SFT training since we already have ample DPO training data. For each sample, we structure the question as a prompt and the chosen response as the target output. Given the considerable training time, we utilize only 50% of this dataset.

4.1.2 M1 DPO Preference Data for DPO

As explained before, it was generated by EPFL Master students in the context of the Modern NLP course. It consists of 26,738 samples with 1,522 unique questions. To ensure data quality and clear preferences, we implement a rigorous cleaning process: We select responses where the preferred option has an advantage, defined as the winning option occurring at least twice as frequently as the losing option. This process aims to enhance the training data quality. We further filter the dataset to include prompts with a length lower than 500 tokens and chosen/rejected responses with a length lower than 750 tokens each. This results in 21,917 samples, sacrificing around 18% of the original dataset for improved quality.

4.1.3 RAG Documents

Choosing the right documents is crucial for improving our model's ability to answer university-level questions. We employ a variety of techniques to develop a comprehensive and suitable compilation of documents as shown in Figure 3.

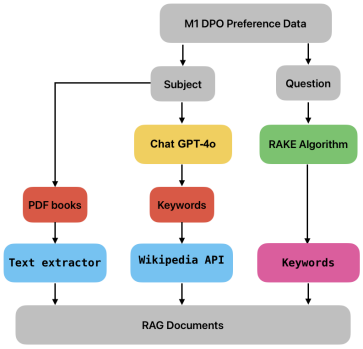

Figure 3: Extracting RAG Documents pipeline

Books for Specific Subjects We search and download several open-source textbooks for each course subject. These books cover a wide range of topics in all areas, offering a diverse and extensive knowledge base. The list of these books can be found in the appendix.

Generate Wikipedia Content Based on Keywords OpenAI's GPT-4o is used to create around 100 keywords for every topic in the M1 DPO Preference Data related to DPO. Specialized terms such as "lemma," "algorithm," and "formula," specific to different areas of study, comprised the list of keywords. We access information from the relevant pages on Wikipedia's API by using these search terms. This process produced approximately 700 keywords with their Wikipedia page content.

RAKE content based on M1 Preference Data questions We utilize a collection of 1,522 distinct questions from the M1 dataset. We employ the RAKE (Rapid Automatic Keyword Extraction) algorithm to determine the most important keywords from these inquiries. RAKE operates by detecting word associations in the text, evaluating the frequency of words appearing together, and deciding on the most important words in the question. This approach yielded 695 unique summary containing around 25 words.

4.1.4 MMLU dataset (Hendrycks et al., 2020) for MCQA and final model evaluation

This dataset comprises MCQs from 57 academic domains. We select the 22 most relevant ones for our project and we format the samples in order to follow the expected format: "subject", "question" which also includes the 4 options, and "answer" which is a single letter from A to D. This gives us a subset of 3,967 samples. This dataset is be used as a benchmark to evaluate the performances of our model with and without RAG.

4.2 Evaluation method

4.2.1 Generator Model evaluation

We employ a quantitative analysis for evaluating our Direct Preference Optimization (DPO) model, focusing on two main tasks: reward computation for preference pairs and multiple-choice question answering (MCQA).For reward computation, we evaluate the model by calculating log probabilities of chosen and rejected responses using policy and reference models, then checking if chosen responses get higher rewards. For MCQA, we assess accuracy by comparing model predictions to correct answers in the MMLU dataset, overall and by subject. In addition to quantitative analysis, we conduct a qualitative analysis to evaluate the relevance of the model's answers. We use a sample of 100 questions to analyze the contextual relevance of the responses. Despite some answers being incorrect (i.e., the model selected the wrong letter), we found that most responses were correct.

4.2.2 RAG Specialization Evaluation

The performance of the RAG model is evaluated based on two criteria: retrieval quality and generated response accuracy. For retrieval quality, we utilize our retrieval pipeline to fetch relevant document chunks for each question. These retrieved document chunks are then assessed for relevance through human evaluation using a sample dataset of 100 questions. For generated response accuracy, we assess the model's accuracy on the MMLU dataset. The evaluation involves generating responses using the RAG-enhanced model and comparing them to the responses generated without RAG to determine the effectiveness of the RAG integration.

4.3 Baselines

We compare our model to its baseline version without any further fine-tuning "unsloth/gemma-7b-bnb-4bit", as well as to other baseline models with equivalent size that are "unsloth/llama-2-7b-bnb-4bit" and "unsloth/mistral-7b-v0.2-bnb-4bit", the quantized versions of Llama 2 7B (Meta, 2023) and Mistral 7B (AI, 2024) respectively.

4.4 Experimental details

4.4.1 Model finetuning:

Our experimental strategy involves two main approaches: (1) DPO-only finetuning and (2) SFT followed by DPO finetuning. We have trained numerous models with various configurations to extensively examine how different hyperparameter settings impact model performance.

- SFT: learning rate lr and warmup steps w
- DPO: learning rate and β, which controls the trade-off between preference alignment and response accuracy to identify the optimal setting that maximizes both the margin and accuracy of the model responses.

For SFT, we tried with $lr = 1e^{-3}$, $w = 5$ and $lr = 5e^{-4}$, $w = 50$, and the training was conducted for one epoch taking, as inputs and labels, the prompts and chosen responses from half of the Argilla dataset respectively.

For DPO, we experimented with multiple values of β (0.1, 0.15, 0.2, 0.5, 0.8) and lr ($1e^{-7}$ and $1e^{-6}$) starting either from the base model that we have chosen or the first SFT model we got previously or the second one. Each DPO training session was run for one epoch due to the repetitive nature of the questions and the extensive training time required. Due to the extensive training duration of about 9 hours on the EPFL SCITAS cluster and to the high numbers of trainings we planned to identify the best parameters, we decided, at first, to train only on the M1 dataset, which we found to be quite large. Then, when we found a food set of hyperparameters ($\beta = 0.8$ and $lr = 1e^{-6}$), we significantly augmented the data by adding the whole Argilla dataset.

4.4.2 RAG pipeline

For the RAG pipeline, we initially implemented a system using sentence transformers such as SBERT (Reimers and Gurevych, 2019) or DistilBERT (Sanh et al., 2019) to generate embeddings and store them in the FAISS database (Johnson et al., 2017). Our goal was to try various embedding models like DistilBERT, SBERT, ALBERT (Lan et al., 2019), and GloVe (Pennington et al., 2014). However, the FAISS database sorting took too much time, preventing us from testing many hyperparameters like chunk size. To improve efficiency, we switched to using LlamaIndex for vector storage with ChromaDB (Team, 2023) and selected the embedding model BAAI/bge-small-en-v1.5 (of Artificial Intelligence, 2023). We experimented with different chunk sizes: 100, 200, 300, and 500.

4.4.3 From sentence to a single letter

After training our model to return sentences, we aimed to refine it to return only a single letter. We explored multiple approaches to achieve this goal.

First, we experimented with generating tokens (trying values between 5 and 50). We, then, compared the generated response to each proposed option using BERTScore, BLEU Score and FuzzyWuzzy, which calculates the similarity between two strings based on the Levenshtein distance. However, the similarities were always too close because the options were too short.

Second, we attempted parsing. We explicitly instructed the model to return a unique letter corresponding to the correct solution, trying different token lengths from 1 to 20. After processing the response to remove special tokens and newline characters, we extracted the first token. If it was either A, B, C, or D, the result was acceptable. Otherwise, we extracted the words until the first newline character and used FuzzyWuzzy to compare with the proposed options. To further refine the model, we fine-tuned it on the ARC dataset, which uses unique letter labels. This aimed to help the model returning a single letter without losing the knowledge but, unfortunately, it did not lead to any improvement.

Both methods mentioned above were also tested using few-shot learning. We provided questions inspired by the M1 dataset, generated by GPT-4o. The idea was to match the MMLU-Benchmark, where the Gemma model is said to achieve the best score with 5-shots. But this was not conclusive.

Finally, we experimented CoT. We, first, let the model answer freely with a brief explanation of up to 80 tokens and, then, take the initial prompt concatenated with the previously acquired answer. The model is only allowed to answer to the second prompt with a single token. To guide it towards the correct path, the final sentence in the prompt is *p = "The correct answer from A, B, C, D is [MASK] "*. The idea is for the model to replace the mask with a single letter. We extract the logits of some target tokens from which we can deduce the predicted letter. This turned out to be a challenging prompt engineering task in order to find the most efficient prompt. We tried to find the best prompt to put before the question

and after the option. This task has been tried in different ways all finishing with *"let's think step by step"* to make it provide a good hint. The latter is, then, concatenated to the previous prompt after replacing the final task with p. Upon examining the most probable tokens, we found that the model often returned the letter within a token containing an underscore or in lowercase. To address this, we calculated the logits for different possible forms of the letters (e.g., for A, we considered A, a, $\_a$, $\_A$). We, then, summed the probabilities over these logits. This technique allowed us to identify the correct letter with greater accuracy. We also tried using the maximum probability and adding a similarity score with the proposed options, but these approaches were not conclusive.

4.5 Results

4.5.1 Model finetuning

Due to lack of space and for sake of clarity, we cannot show all the plots that we acquired during the training of the different models. However, we present below two graphs comparing the evolution, for the validation dataset of the accuracy, which indicates the percentage of correct preferences, and the margin, which measures the confidence of the model in preferring one response over another. In these graphs, only the plots of the models that performed best in each finetuning method are given.

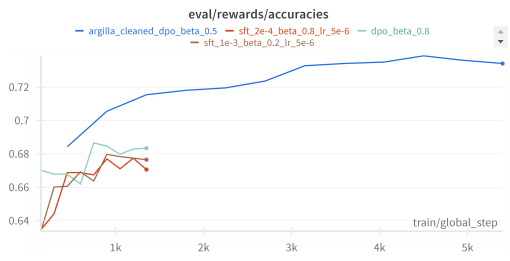

Figure 4: Accuracy comparison

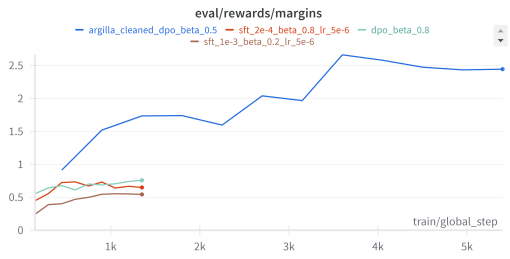

Figure 5: Margin comparison

The results show a clear superiority of the model that finetunes the base model directly by applying dpo to the large dataset consisting of the M1 preference dataset and the whole argilla dataset. Therefore, this model is chosen as our final model and will be compared with the baselines.

4.5.2 Preference dataset generation

Upon evaluating the answers we got from both strategies we tried, we found that, surprisingly, the second method implementing CoT provided less interesting answers compared to the standard method. This can be explained by the fact that it adds noise to the question. If the initial answer is incorrect, there's a very small chance of recovery and even if the hint is good, it might introduce additional information that confuses the model and leads to an incorrect answer

4.5.3 Rag Retrieval quality

The retrieval quality has been controlled by us looking into a subset of approximately 100 questions. The vast majority seems to be relevant to the query providing valuable information that can help to answer the question. An example is provided in appendix.

4.5.4 Comparison with baselines:

In the following table, we detail the performance of different models (Edu being our final model) using the MMLU Benchmark. We only kept the subjects in relation with STEM for the final MMLU dataset. We, also, give their performances over key subjects that are the ones targeted by our chatbot.

Subject	**-**	**Edu**	**Gemma**	**Mistral**	**Llama2**
Algebra	base	**34**	34	24	21
	rag	**36**	30	28	23
CS	base	**48**	47	46	24
	rag	**42**	42	32	23
Math	base	**29**	28	25	20
	rag	**32**	28	30	23
Physics	base	**40.2**	34,3	34,3	20.5
	rag	**41.2**	31.4	24,5	21.5
Comp	base	53	52	**57**	30
Security	rag	**54**	47	53	29
EE	base	**54.5**	54.4	40	24
	rag	**51.7**	46.2	39,3	24
MMLU	base	**47**	37.5	38.8	37.5
	rag	44.4	36.4	36	35.2

Table 1: Performance Metrics

5 Analysis

5.1 Model finetuning

Two interesting points can be noted in this section:

Comparison between the 2 strategies: Comparing the best-performing DPO-only model with the best-performing SFT + DPO model that have been both trained on the M1 preference dataset with $\beta = 0.8$, we observed that the DPO-only model outperformed the SFT + DPO model in terms of both accuracy (0.683 vs 0.671) and margin (0.76 vs 0.65). This suggests that, unexpectedly, the SFT phase might not contribute positively to the model's performance and could potentially hinder it. According to some sources (DeepMind, 2023), the best DPO model is not necessarily achieved with the best SFT model. A possible reason for our unexpected result is that the SFT was trained with hyperparameters that do not suit the subsequent DPO training. Due to time constraints, we could not try different configurations to optimize the SFT.

Comparison between 2 models with the same parameters but different datasets The DPO model trained with the additional Argilla dataset, which is significantly larger than the dataset we used before, showed improved performance. This is rational for 2 reasons. First, our model is rather complex, so it needs to be trained on a substantial dataset to achieve optimal performance. This finding aligns with the concept of scale laws in large language models (LLMs), which suggest that model performance improves predictably with an increase in the size of the training data and the number of model parameters. Second, we notice that for the models trained on the M1 dataset quickly reached their peak performance, after which further training did not lead to improvements regardless of the learning rate. This observation suggests that, apart from the scale law, the M1 dataset quality might not be sufficient to drive significant improvements. We can conclude that asking students, who are not experts in all the proposed domains, to generate a preference dataset leads to a very noisy dataset. This noise likely hindered the training of our model, making it difficult to achieve optimal performance. The inconsistency and potential inaccuracies in the preference data could have contributed to the observed performance limitations.

5.2 RAG system

Our analysis of the Retrieval-Augmented Generation (RAG) system revealed key insights into its performance and optimization. Testing with different numbers of chunks (1, 2, and 3) on a test dataset from the MMLU dataset across various subjects, we found that the top 3 chunks were relevant for over 90% of the queries. This indicates the system's effectiveness in fetching pertinent context, with a chunk size of 500 characters offering an optimal balance between relevance and computational efficiency.

Qualitative analysis confirmed the retrieval process did not bottleneck inference. We observed significant performance variations with different prompt phrasings. The prompt "Given the context information and not prior knowledge, answer the query" produced contextually relevant but often general responses. Conversely, "Context information is below" prompted the model to effectively use the provided context and its prior knowledge, resulting in more precise answers. Chain-of-Thought (CoT) prompting was also crucial for maintaining accuracy.

5.3 Analyzing the table of performances

Noteworthy aspects emerge while reviewing the table 1.

Comparison with baselines: Our model GemmaEdu clearly stands out as itl consistently outperforms the other models (Gemma, Mistral, and Llama2) across all the categories. This demonstrates the effectiveness of our model's training and signifies a substantial improvement in performance.

Impact of RAG: Focusing on our Edu model, it is worth noting that the RAG setting does not always lead to improved performance compared to the base setting. In fact, in the overall MMLU, the RAG setting results in lower accuracy, which may suggest that the chunks provided are not as helpful as expected, despite the human evaluation performed. However, upon closer examination, we can see that the RAG setting outperforms the base setting in most categories, with the exception of CS. This exception can be attributed to the fact that our model was primarily trained on CS data, and the additional context provided in the RAG setting may introduce noise that negatively impacts the model's performance. Therefore, we can observe a Simpson's paradox in this case, where the overall

trend does not necessarily reflect the trends in individual categories. This highlights the importance of considering both the overall performance and the performance in specific categories when evaluating our model.

Analyzing performance: While our Edu model outperforms the other models, it is important to acknowledge that the overall accuracy of 47% in the MMLU dataset, although better than random, still leaves room for improvement. Interestingly, human evaluation conducted by our team reveals that the hints provided to the model often contain the correct answer, indicating that the model has effectively gained knowledge. However, the requirement for the model to return only a single letter as the answer appears to hinder its performance. This limitation could be attributed to either the prompt provided to the model or the model itself, as it has not been specifically trained to return a single letter as an answer. Our attempts to fine-tune the model on the ARC dataset to improve its ability to return a single letter did not result in any significant performance improvements which suggest possible improvements in the prompt especially that we are far from the value given in the *Huggingface* documentation suggesting an accuracy of 64.3% for the original Gemma-7b model.

6 Ethical considerations

To adapt our model for high-resource languages (e.g., French, German), we could finetune it on large multilingual datasets, leveraging cross-lingual embeddings and domain-specific data. For low-resource languages (e.g., Urdu, Swahili), we could apply transfer learning from high-resource languages, use data augmentation techniques like back-translation and data synthesis, and employ unsupervised or semi-supervised learning with large amounts of unlabeled text. We should also ensure that our RAG setup includes a diverse multilingual knowledge base and supports multilingual retrieval mechanisms.

To adapt our model for interaction with users in signed languages, we can collect and preprocess datasets of signed language videos with corresponding MCQ content and answers. We can use computer vision techniques for sign language recognition and pose estimation, and develop 3D avatars or generative models for sign language video synthesis. The model can be trained to translate text-based MCQs into sign language and recognize signed responses. We can integrate real-time sign language translation and interactive interfaces supporting video input. Feedback from the Deaf community are essential to ensure accuracy and cultural sensitivity.

If our model functions as intended, students majoring in scientific fields will benefit most, as the AI chatbot will provide accurate answers to Multiple Choice Questions in these subjects. However, if misused, it can cause harm to both students and teachers. For example, the chatbot might deliver incorrect information, assuming that a correct answer is always present among the provided options. Additionally, if prompted with malicious questions, it could leak sensitive information, compromising the privacy of dataset authors. The chatbot also poses risks if used for harmful purposes.

Vulnerable and marginalized individuals are especially at risk from such misuse, as they may feel safer talking to a chatbot than to people in real life. To minimize these risks, our model can be further fine-tuned using DPO to detect potentially harmful inquiries and respond in a safe manner.

Finally, note that our model and datasets cannot be publicly published because some documents used for RAG extension are protected by copyright law. Written permission is required for any future publication of our work.

7 Conclusion

In this project, we successfully developed an advanced educational assistant chatbot using the quantized Gemma 7B model. Our primary findings include learning how to fine-tune large language models (LLMs) using Self-Supervised Fine-Tuning (SFT) and Direct Preference Optimization (DPO) and effectively implementing them. Additionally, we incorporated Retrieval-Augmented Generation (RAG) and improved prompt engineering techniques to boost the model's effectiveness and obtain the desired outputs. Our results demonstrate that our fine-tuned model consistently outperforms baseline models, indicating significant improvements in accuracy and relevance. However, we faced limitations such as not achieving the expected benchmark performance, possibly due to not testing over the entire MMLU dataset and focusing on specific subjects. For future work, we with extending the RAG database, incorporating more documents, and exploring advanced prompt engineering techniques to further enhance response quality.

References

Mistral AI. 2024. Hugging face - mistral.

Alvaro Bartolome, Gabriel Martin, and Daniel Vila. 2023. Notus. https://github.com/argilla-io/notus.

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

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. 2023. Ultrafeedback: Boosting language models with high-quality feedback.

DeepMind. 2023. Aligning llms with direct preference optimization. Accessed: 2023-06-04.

Google. 2024. Hugging face - gemma.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Nan Jiang, Xiaopeng Li, Shiqi Wang, Qiang Zhou, Soneya Binta Hossain, Baishakhi Ray, Varun Kumar, Xiaofei Ma, and Anoop Deoras. 2024. Training llms to better self-debug and explain code. *arXiv preprint arXiv:2405.18649*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Faiss: A library for efficient similarity search and clustering of dense vectors. *arXiv preprint arXiv:1702.08734*.

kagglepro. 2024. Medium.

Sungdong Kim, Sanghwan Bae, Jamin Shin, Soyoung Kang, Donghyun Kwak, Kang Min Yoo, and Minjoon Seo. 2023. Aligning large language models through synthetic feedback. *arXiv preprint arXiv:2305.13735*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Meta. 2023. Hugging face - llama 2.

Beijing Academy of Artificial Intelligence. 2023. Baai/bge-small-en-v1.5. https://huggingface.co/BAAI/bge-small-en-v1.5.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

ChromaDB Team. 2023. Chromadb: A high performance vector database. https://www.trychroma.com/.

unsloth. 2023. Gemma-7b. Accessed: 2024-06-04.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models.

A Appendix

A.1 Contributions

- Salim Boussofara: M1 Preference data collection, model finetuning with different configurations, letter extraction, report writing.
- Oussama Gabouj: M1 Preference data collection, RAG documents, RAG pipeline, prompt engineering, report writing.
- Yasmine Chaker: M1 Preference data collection, additional data collection, data preprocessing, RAG documents, evaluation, report writing.

A.2 Books for RAG

A.2.1 Artificial Intelligence and Machine Learning books

1. *Machine Learning* by Tom M. Mitchell | 1997 | McGraw-Hill Science/Engineering/Math
2. *Explorations in Artificial Intelligence and Machine Learning* | CRC Press
3. *Artificial Intelligence: A Modern Approach* by Stuart Russell and Peter Norvig | 2021 | Pearson
4. *Introduction to Machine Learning* by Alex Smola and S.V.N. Vishwanathan | 2012 | Cambridge University Press
5. *Artificial Intelligence: Machine Learning and Deep Learning* by Oswald Campesato | 2020 | Mercury Learning and Information
6. *Mathematics for Machine Learning* by Marc Pete Deisenroth, A. Aldo Faisal and Cheng Soon Ong

A.2.2 Computer Science Theory books

1. *Theory of Computation Lecture Notes*- Subject Code: BCS-303 | Bachelor of Technology in Computer Science and Engineering & Information Technology
2. *Theory of Computer Science: Reductions* by Gabriele Röger | 2021 University of Basel
3. *The Role of Theory in Computer Science* by John E. Savage

A.2.3 Computer Software books

1. *Chapter 4: Computer Software* by Dr. Rahman Ali | University of Peshawar
2. *Introduction to Computer Software* by BASNAYAKA W.B.M.C.M. | SUBJECT CODE: HNDA1105

A.2.4 Computer Systems books

1. *Computer System Architecture, Third Edition* by M. Morris Mano | International Edition
2. *Introduction to Computing Systems: From Bits & Gates to C/C++ & Beyond* by Yale N. Patt, Sanjay J. Patel
3. *BCA-121 Computer Fundamental*
4. *Computer Systems: A Programmer's Perspective, Third Global Edition* by Randal E. Bryant, Carnegie David R. O'Hallaron, Carnegie | Mellon University

A.2.5 Electrical Engineering books

1. *Electrical Engineer's Reference Book*, Sixteenth edition by M. A. Laughton CEng., D. J. Warne CEng., | FIEE
2. *The Electrical Engineering Handbook* by Wai-Kai Chen
3. *Introduction to Electrical Engineering* by Mulukutla S. Sarma | Oxford University Press
4. *Handbook of Electrical Engineering for Practitioners in the Oil, Gas and Petrochemical Industry* by Alan L. Sheldrake | Consulting Electrical Engineer, Bangalore, India

A.2.6 Theoretical Physics books

1. *Basic Theoretical Physics* by Uwe Krey, Anthony Owen | Springer
2. *100 Years of Fundamental Theoretical Physics in the Palm of Your Hand: Integrated Technical Treatment* by B. Manoukian
3. *Theoretical Physics 6: Quantum Mechanics - Basics* by Wolfgang Nolting
4. *Theoretical Physics 1: Classical Mechanics* by Wolfgang Nolting

A.2.7 Abstract Algebra book

1. *Abstract Algebra Theory and Applications* by Thomas W.Judson and Stephen F. |Austin State University

A.3 Finetuning evaluation

A.3.1 DPO Finetuning evaluation

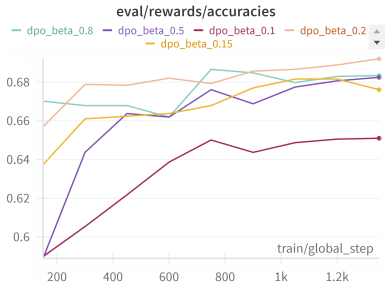

Figure 6: DPO Accuracy evaluation

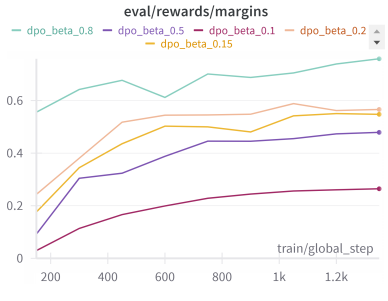

Figure 7: DPO Margin evaluation

A.3.2 SFT DPO Finetuning evaluation with $lr = 1e-3, w = 50$

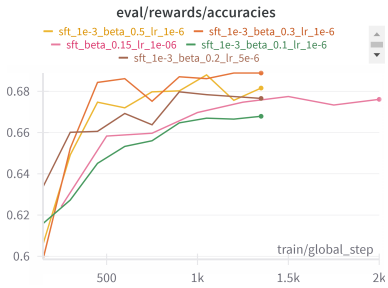

Figure 8: Accuracy comparison

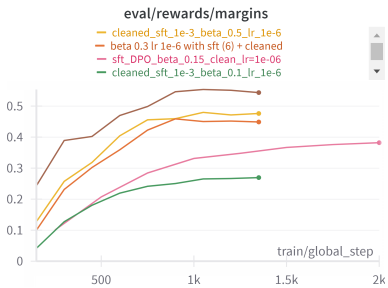

Figure 9: Margin comparison

A.3.3 SFT DPO Finetuning evaluation with $lr = 2e-4, w = 5$

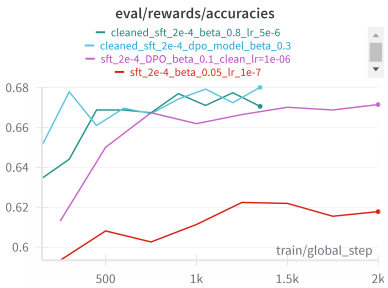

Figure 10: SFT + DPO Accuracy evaluation

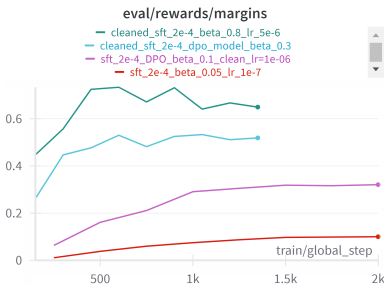

Figure 11: SFT + DPO Margin evaluationn

A.3.4 SFT DPO Finetuning overall evaluation

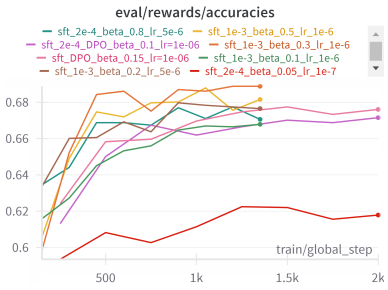

Figure 12: Accuracy comparison

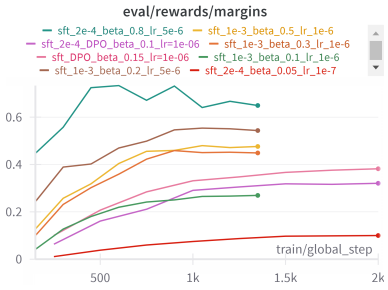

Figure 13: Margin comparison

A.4 RAG retrieval example

Query:

Question: Suppose that a certain software product has a mean time between failures of 10,000 hours and has a mean time to repair of 20 hours. If the product is used by 100 customers, what is its availability?
Options: A. 80% B. 90% C. 98% D. 99.80%

Retrieved chunk:

=== Availability === Availability is the fraction of time during which the system can respond to requests.