Final Report: CheatSheet: A Generator Model for Academic MCQ Answering

Baptiste Maquignaz | 312360 | baptiste.maquignaz@epfl.ch
Adam Ben Slama | 314806 | adam.benslama@epfl.ch
Mahmoud Dokmak | 301995 | mahmoud.dokmak@epfl.ch
Garik Sahakyan | 314372 | garik.sahakyan@epfl.ch
Fantastic Four

Abstract

This report presents "CheatSheet", a causal model for text generation that was designed to tackle the most challenging university-level exam. The name "CheatSheet" is inspired by the traditional two-sided cheat sheets allowed by teachers to students to help them during their EPFL exams.

In the following pages, we describe our efforts to develop a custom-tailored AI with limited computational resources, time, and experience, within the constraints of a student course project. This AI aims to assist EPFL students in their learning journey. We detail the adaptation of a GPT-2 large model to enhance its reasoning capabilities specifically for answering complex exam questions that require deep scientific comprehension and knowledge.

In the initial phase of our study, we focused on fine-tuning CheatSheet to effectively address a broad spectrum of both open and multiple-choice science-related questions. To enhance the quality and clarity of the responses, we utilized human feedback through Direct Preference Optimization (DPO) to ensure the answers were understandable, well-structured, and complete from a human perspective. In the second part of this project, we restricted the objective to answering multiple choice questions (MCQ) with a single letter. We performed quantization as well as Retrieval Augmented Generation (RAG) in an attempt to augment our model's performance by either reducing its size or augmenting its knowledge. Our research prove that our model slightly outperforms the large GPT-2 model, even when reducing the model's size.

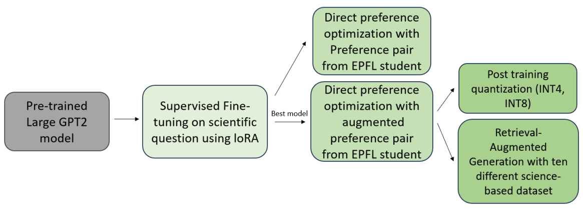
1 Introduction

At the time of writing this report, the advent of artificial intelligence (AI) has profoundly transformed our society. The automation of various tasks has streamlined processes, raising questions about the utility of traditional skills. With this paradigm shift rises a critical need of rethinking our ways of learning. For aspiring engineers like us, access to advanced chatbots has almost become essential. These intelligent systems can clarify concepts, provide complementary knowledge, summarize complex subjects, and answer questions, significantly enhancing our learning experience. However, the use of AI has also introduced numerous challenges. Issues of transparency and risk of exposure to fallacious yet convincing reasoning have become increasingly prominent.

Despite the impressive capacity of Large Language Models (LLMs) to provide information to users in a natural way, widely used chatbots often struggle with answering complex problems in specialized fields due to their broad training scope. As engineering students, we encounter the limitations of these chatbots in advanced scientific knowledge on a daily basis. As underlined in multiple studies such as (Hendrycks et al., 2021), AI often prove deceiving in applied mathematics, physics or even coding problems, making basic arithmetic mistakes or providing wrong arguments. Our project aims at improving the scientific knowledge and reasoning capability of a LLM, making it capable of answering complex EPFL exam questions.

In this project, we started from a GPT-2 (Radford et al., 2019) large model that we fine-tuned for the question answering task. We used a variety of mathematics, physics, machine learning and coding questions/answers datasets to enhance the reasoning capabilities of our rough model for basic science related tasks. In an effort to significantly improve our model's performances and fine tune it for EPFL question answering, the students of the class created a custom preference dataset derived from the material of more than 100 EPFL courses. A pair of solutions were generated by ChatGPT 3.5 (Brown et al., 2020) and evaluated by the students to allow Direct Preference Optimization (DPO), hopefully reinforcing the model's capacity to provide highly qualitative answers.

In the second part, we focused on answering multiple-choice questions with a single letter response. To optimize our model's performance, we augmented Cheatsheet using Retrieval-Augmented Generation (RAG), enabling the model to retrieve information from external documents. Additionally, we attempted to reduce the model's size to decrease inference time and energy consumption, while minimizing any potential decline in answer quality.


Pre-trained Large GPT2 model
Supervised Fine-tuning on scientific question using loRA
Best model
Direct preference optimization with Preference pair from EPFL student
Direct preference optimization with augmented preference pair from EPFL student
Post training quantization (INT4, INT8)
Retrieval-Augmented Generation with ten different science-based dataset

It is important to note that the primary goal of this work is academic, aimed at gaining experience in Natural Language Processing (NLP). Given various constraints such as computational and financial resources, time, and our very limited experience, achieving state-of-the-art results was not feasible. Instead, the true purpose of this project was to explore the extent of our capabilities and see how far we could progress.

2 Related Work

Prompting ChatGPT 3.5 to obtain a dataset

Collecting preference pairs using a chatbot is a well-established pipeline, and numerous studies have explored optimal ways to obtain the best answers from a pre-trained LLM. (Lester et al., 2021) were among the first to demonstrate the power of prompting for task adaptation of pre-trained models. The prompt structure and pattern have a significant impact on how the model comprehends the task. We were inspired by the work of (White et al., 2023) to structure our prompt in an efficient way. As introduced in (Wei et al., 2022), Chain of Thought (CoT) prompting encourages complex multi-step reasoning by guiding the model through step-by-step answers, thereby conditioning frozen language models to perform specific downstream tasks. Inspired by (Kojima et al., 2022) we leveraged this CoT reasoning process by using the triggering sentence "Let's think step by step".

Fine-tuning a pretrained LLM for a downstream task

Current LLMs such as GPT-3 (Brown et al., 2020), LLaMa (Touvron et al., 2023) or RoBERTa (Liu et al., 2019) were trained to achieve a wide variety of tasks and subjects with a single model. However, it is often needed for a model to perform on a single downstream task. To fine-tune a model on a dowsntream tasks, many strategies can be used. Prompt-tuning is an effective way to adapt a pre-trained LLM for the downstream task. However, this method can prove sensible to slight changes (Liu et al., 2023). Supervised Fine-Tuning involves further training the model on a smaller set of high-quality data specifically tailored for the target task. This process allows the model to specialize in the desired field and achieve greater performance in the desired task (Dodge et al., 2020). Direct Preference Optimization (DPO) directly trains models to learn from user preference feedback to enhance output relevance and quality. The use of human feedback to reinforce certain behavior of the model that were judged positively by users has emerged as a new method to increase performance of a model. We followed the methodology of (Rafailov et al., 2024) by first fine-tuning our model using SFT and then further optimizing its responses using DPO.

LLMs for MCQ answering

LLMs such as GPT-2 is known to be particularly sensitive to prompting. We tried to fight this limitation by using Proportion of Plurality Agreement (PPA) (Robinson et al., 2023) to measure the model's confidence in its MCQ answer and output the best one.

3 Approach

This section contains the complete pipeline of training CheatSheet and a detailed description of our methodology. We started from a GPT-2 large model and fine-tuned it on a wide variety of science questions to then further encouraging high quality response using DPO. After further prompt engineering to specialize CheatSheet in the task of MCQA, we performed a quantization of the model as well as RAG in an attempt to further improve it.

3.1 Base Model Choice

The base model of CheatSheet is a GPT-2 large model. GPT-2, released in 2019 by OpenAI, is a competitive decoder-only model that achieved state-of-the-art results on numerous language modeling datasets. The model is highly efficient for tasks such as text generation, summarization, and translation. Initially, we considered using Flan-T5 (Chung et al., 2024). These models outperform prior public checkpoints such as T5 and have strong zero-shot, few-shot, and Chain of Thought (CoT) abilities. However, due to our limited experience in the NLP field, we opted for a heavily documented model and felt more comfortable with training a decoder-only model.

3.2 Supervised Fine-Tuning (SFT)

After choosing our base model, we started training it on a wide variety of both open and MC question datasets (see Section 4.1) to adapt its behavior to the desired task. To fit our restrained resources and time constraints, we used a LoRA (Hu et al., 2021) configuration to reduce the number of parameters to train.

3.3 Collecting Preference Data

Each student of the course had to use a ChatGPT 3.5 wrapper to answer EPFL courses' questions. Those questions can prove extremely difficult even for master students and in an attempt to get answers that were as qualitative as possible we used several techniques. As every author processed in slightly different way, we will relay the main findings to design our prompts.

- We used the triggering sentence "Let's think step by step" as it was proven (Kojima et al., 2022) to enhance the CoT capability of LLMs, improving the quality of the response greatly.

- We attempted few-shot prompting to provide the model with several examples of questions before answering. However, we observed that this approach led to lower quality answers because the subjects varied so much that it was impossible to find relevant examples for all questions, ultimately misleading the model.

- Each question had a corresponding course ID that we mapped to a scientific field. To achieve this, we sorted all the questions by course ID and used ChatGPT to guess the topic. We then incorporated this topic into the prompt to guide the model towards the correct subject area.

- To generate a second answer while maintaining quality, we considered two main approaches: writing a different prompt to elicit another high-quality answer or feeding the previous answer back into ChatGPT, asking it to either correct or rephrase it. The former approach resulted in more varied but generally lower quality answers, while the latter approach produced more consistent and higher quality responses. This trade off between variety and quality was one of the key aspect to optimize.

You can find an example of prompt that we used to obtain preference pairs in A.2).

3.4 Direct Preference Optimization (DPO)

DPO directly leverages user feedback to improve model performance and output quality. By focusing on user preferences, DPO ensures that the model's outputs align more closely with user expectations and needs, leading to a better user experience. Following a similar pipeline as (Rafailov et al., 2024), we used the preference dataset built by EPFL student to further train the fine-tuned model. We again used LoRA to reduce computational complexity.

3.5 Specializing for MCQ Answering

In the second part of the project, we specialized our model in answering with a single letter. We had several ideas on how to perform this modification:

- Training a separate model to classify CheatSheet's answers into four categories (A, B, C, or D). To achieve this, we intended to replace the questions in an MCQ dataset with the responses from our model. The additional model would then be trained to map these responses to a single letter. However, this approach had a significant flaw: if CheatSheet provided an incorrect answer, the classifier model would learn to map that incorrect answer to a letter, potentially perpetuating errors.

- Further training CheatSheet to write a single letter. We could re-train our model on other MCQ datasets to learn this particular task. However, we think that re-training the model on this new task would completely overwrite the learned behaviors when performing SFT and DPO.

- We settled on prompt engineering to simply reuse the weights of CheatSheet without further training. However, GPT-2 is highly sensible to prompting and even a change in a single word can lead to completely different behaviors. To improve robustness, we shuffled the order of the options and fed every single permutation to the model as in (Robinson et al., 2023). We then output the most answered option (see 1).

3.6 Retrieval Augmented Generation Specialization

The first step of RAG was to search scientific documents which the model could use to answer its given questions. This would allow the model to not only rely on its own knowledge, but to also have a database from which it could retrieve some useful information. To this purpose, we collected many datasets from Hugging Face consisting scientific facts, as well as some QA datasets of different topics including mathematics, quantum mechanics, thermodynamics, biology and machine learning. Once we had our dataset ready, our initial idea was to use the Hugging Face RAGModel, because they created an entire ready to use RAG pipeline. However, their implementation worked only with sequence to sequence generators, and since our chosen generator was a causal LM, we needed to create our own pipeline. To create this retriever pipeline, we first used a pre-trained encoder to compute the embedding of each entry in our constructed dataset. We then encoded our question, and compared its embedding to each encoded entry of the dataset. We retrieved the $top - k$ entries of the dataset that were the most semantically similar to our question. These entries were then concatenated to our question, serving as context to our model, and were passed through the MCQ pipeline described in section 3.5

3.7 Quantization

Regarding quantization, while Quantization Aware Training (QAT) was an option, we decided to implement Post-Training Quantization (PTQ) for Milestone 3. The primary objective for this milestone was to achieve a model that is lighter than the one developed in Milestone 2. We explored various types of quantization and selected both INT4 and INT8 quantization methods. This approach allowed us to produce three different models for comparison, facilitating a comprehensive evaluation of their performance, efficiency and limitations. Practically, we used the integration of Hugging Face's Transformers library with the Bitsandbytes library. Bitsandbytes is a well-known quantization library that offers state-of-the-art techniques for reducing the memory footprint of models without significantly sacrificing performance. This library allows us to choose the bit precision for loading the model(which is a basic quantization technique), but it also lets us modify the data type used during computation which is an interesting feature to speed up inference time. With the help of the (Rajpurohit, 2023) article on BitsandBytes, we were able to apply these techniques and experiments with different optimizations to determine which works best for our model and which do not.

4 Experiments

4.1 Data

- **SFT:** we selected two math-based datasets, OrcaMath (Microsoft, 2023) and MetamathQA (MetaMath, 2023), combining basic math question/answer pairs to enhance our model's chain of thought for basic mathematical reasoning. To ensure our model possesses fundamental physics knowledge, we incorporated a physics dataset (ArtifactAI, 2023). Lastly, we used two coding datasets, Glaive Code Assistant (GlaiveAI, 2023) and TinyCodes (AI, 2023), to improve our model's coding capabilities, acknowledging the complexity and unpredictability of coding tasks compared to text generation.

- **DPO:** In order to perform DPO, we used the preference data collected by the students of the class and the same dataset augmented with data from DistilLabel (Argilla, 2023). PvDPO (Durbin, 2023) and PRHedge (Hegde, 2023) which contain python and mathematics based questions.

- **MCQA:** In order to assess the accuracy of our model in answering MCQs, we needed a dataset that reflected the difficulty levels of EPFL questions. We filtered our test dataset from EPFL to specifically retain MCQs and proceeded by replacing the preference pair with the actual answer letter using a ChatGPT-3.5 wrapper. We further refined the dataset by

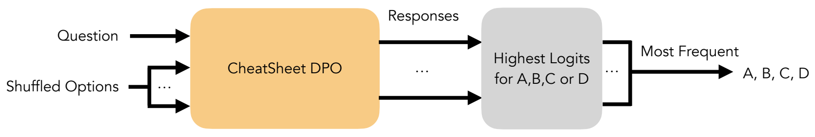


Figure 1: Pipeline for MCQA

retaining only the questions with four answer choices.

- **Retrieval-Augmented Generation (RAG):** As explained above, we need some datasets from which our model could extract some context. We decided to search for other datasets than the ones used in the SFT/DPO training part, since we considered that our model had already learned these datasets. We used the Hugging Face datasets consisting of mathematics (Kenney, 2023), physics (Ayoubkirouane, 2023), (jilp00, 2023a), (jilp00, 2023b), (jilp00, 2023c), machine learning (mjphayes, 2023), (Mridul-Dixit, 2023), (whiteOUO, 2023) and others (Li et al., 2023), (Clement et al., 2019). These choices were motivated by the fact that such topics are very common in EPFL exams.

4.2 Evaluation method

Measuring performance of a text-generating model is a tough task because the quality of text is extremely subjective and the wanted properties differ from task to task.

SFT

When performing SFT, we want our model's answer to be 'close' to the actual answer of the problem. To measure this similarity between our answer and the provided one, we used several metrics:

- **BLEU** (Papineni et al., 2002) measures how closely the generated text matches one or more reference texts based on n-gram overlaps. Higher scores indicate similarities in sub-sequences between the target and the output.
- **ROUGE** (Lin, 2004) measures the overlap of n-grams, word sequences, and word pairs between the generated text and reference texts. Key variants include ROUGE-N (n-gram recall), ROUGE-L (longest common subsequence), and ROUGE-S (skip-bigram).
- **BERTScore** (Zhang et al., 2019) compares contextual embeddings from BERT (Devlin et al., 2018) of the target and output using cosine similarity. It captures semantic similarity more effectively than 'naive' metrics like BLEU and ROUGE.

DPO

In DPO, the goal is to obtain a model that will choose an answer over a less qualitative one and therefore requires different metrics. To evaluate the model's ability to select the correct sentence in a preference pair, we computed the policy reward accuracy of the model across its key steps. This metric roughly measures the proportion of instances where the reward margin of the policy model exceeds that of the base model. Namely, the proportion of time where the DPO trained model makes a bigger difference between the chosen or rejected sentence. We measured policy reward accuracy within the EPFL students' dataset.

We thought about computing different metrics such as the ones for SFT and compare the different scores for both the chosen and rejected answer. However since the dataset contains a lot of pairs that are extremely similar contextually and semantically, we could not draw any conclusion from those metrics.

We chose to include only EPFL questions in our test set, as we believe this best represents the final task: answering questions from EPFL courses. It is important to note that other datasets containing mathematics or physics problems are often significantly simpler than those from EPFL, making them less relevant for our purposes.

MCQA, RAG, Quantization

Answering MCQ with a single letter is again a classification task where we want to measure the accuracy of the model's response. Again we only used MCQs from the EPFL dataset to measure the performance of the model because the difficulty of the questions stand out compared to any scientific dataset.

4.3 Baselines

As a baseline throughout the different steps, we used the pretrained GPT-2 model. This logical choice allowed us to compare the impact of every step of the training process. During MCQA, only keeping the 4-answer questions allowed us to establish an additional straightforward baseline of exactly 25%, corresponding to random guesses.

4.4 Experimental details

Hyperparameters and set up: During SFT we used the standard cross-entropy loss on Q&A datasets, utilizing the existing HuggingFace `SFTTrainer` class. We trained the model using the LoRA configuration for GPT-2. After testing different values of hyperparameters, we found that a rank $r = 16$ and scaling factor $\alpha = 32$ led to good performance while corresponding to our restricted resources. This set up lead to around 3M unfrozen parameters for training. Since we are performing finetuning on a pretrained model, we opted for a small dropout rate of 0.1 as the model should already prove quite robust.

During DPO training, we used the standard DPO loss [A.1] using the HuggingFace `DPOTrainer`. Following the recommendations from the Alignment Handbook by (Tunstall et al., 2023), we opted for bigger rank $r = 128$ and scaling factor $\alpha = 128$ with smaller dropout rate of 0.05. This configuration lets roughly 30M parameters to train. We tuned the β scaling parameter in the DPO loss [A.1] by testing different values ranging from 0.01 to 1, and found that $\beta = 0.35$ yielded the best policy reward accuracy.

For RAG, we used a Hugging Face pre-trained question encoder and context encoder (Karpukhin et al., 2020). They have been trained on wiki-dpr (Karpukhin et al., 2020) which is a huge dataset consisting of 20M passages from Wikipedia. We believe that this model gives us good encoding due to its huge and diverse entries. We further indexed our encoded dataset using the Faiss library to have faster document retrieval. We tested our RAG model on its ability to answer MCQ questions by retrieving k entries from our dataset for $k \in 1, 2, 3$.

Learning rate: To ensure a meaningful learning rate in both SFT and DPO, we used a cosine scheduler with a warm-up phase of 0.1 along with an Adam optimizer. We found that an initial learning rate of $2 \cdot 10^{-4}$ lead to an efficient training for SFT. For DPO, we concluded that a smaller initial learning rate of $6 \cdot 10^{-6}$ worked out well.

Training time:

All training, including Supervised Fine-Tuning (SFT) and Direct Preference Optimization (DPO), utilized a single A100 GPU. SFT required 15 hours to train the model over 3 epochs on a dataset comprising 350k question-answer pairs. To reduce the needed memory and avoid losing considerable amount of time in case of a problem, we split this dataset in two. This also allowed to measure how much data were necessary to reach an optimal policy. DPO, conducted over 3 epochs with approximately 50k preference pairs, was completed in 5 hours. This training duration contrasts sharply with the months-long timelines typically required for training large models from scratch on extensive parallel GPU arrays. Given our constrained schedule and low compute resources, achieving more extensive training was challenging.

Results:

For SFT, we trained our base model on different Q&A datasets for a total of 350k questions. Half dataset and full dataset indicate that the model have been trained either on half of the dataset or the full dataset (as explained in 4.4). As we can see in 2, the training loss, which closely follows the evaluation loss, is rather stable during the second half of training, implying that the model had not learned much more with the additional 175k data.

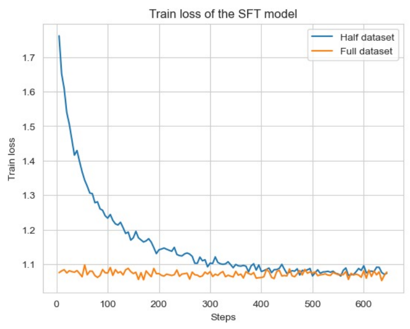


Figure 2: Training loss during SFT on both half of dataset data

We obtained encouraging results (3) that show how the model has been finetuned to answer basic scientific questions. Our two models outperform the base GPT-2 on all datasets and metrics, with one exception: the BLEU score on TinyCodes.

We performed DPO with solely the student pref-

F1 BLEU score	ARXIV-PHYSICS	GLAIVE-CODE	META-MATH	ORCA-MATH	TINY-CODES
Base Model	0.287	0.177	0.266	0.270	**0.086**
50% SFT	0.296	0.196	0.383	**0.344**	0.043
100% SFT	**0.297**	**0.198**	**0.387**	0.339	0.044
F1 ROUGE-l score					
Base Model	0.176	0.180	0.180	0.188	0.108
50% SFT	**0.194**	0.216	**0.332**	**0.309**	**0.138**
100% SFT	0.191	**0.226**	0.328	0.294	0.136
F1 BERTScore					
Base Model	0.8211	0.800	0.825	0.815	0.760
50% SFT	**0.8289**	0.828	**0.860**	**0.853**	0.825
100% SFT	0.8287	**0.830**	0.861	0.851	**0.828**

Figure 3: Results after SFT

erence pairs and then with augmented data (see 4.1). When looking at the evolution of rewards ((4), we can see that the margin between the reward given to the chosen and rejected sentence increases throughout training and seem to stabilize at the end, indicating that the training reached stability.

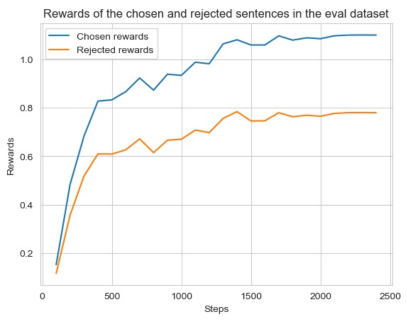


Figure 4: Evolution of rewards of test dataset chosen and rejected sentence throughout training

We then measured the policy reward accuracy on a test set composed of unseen data from the EPFL dataset and obtained our first surprise: the highest accuracy is achieved by the model trained solely on the EPFL preference dataset.

Policy Reward Accuracy	EPFL Dataset [%]
CheatSheetSFT	54.87
CheatSheetDPO	**55.51**
CheatSheetDPOaug	53.89

Table 1: Accuracy across different step of fine-tuning and training

Specializing the model for MCQA, we measured the accuracy of *CheatSheetDPOaug* as well as the base pretrained model. We obtained curious results 2. All the model variations seem to achieve particularly low performance when answering EPFL MCQ. In fact, the base model seem to have a lower accuracy than the random baseline of 25%. After reflexion, we concluded that this could be due to several factors. Firstly, the test set is relatively small, consisting of approximately 500 questions, which may introduce variability due to statistical anomalies. Secondly, many questions, particularly those from exams, are deliberately designed to include traps. This is the reason why we learned as students not to answer an exam MCQ when in doubt, even though the expectation when filling an entire exam at random would be zero. Concerning RAG, the two versions are different by the number of entries we retrieved from the RAG dataset to use as context for our prediction. We can see that by retrieving 2 entries results in higher accuracy than by retrieving only one. Moreover, we notice that RAG improves the overall accuracy of the model. Finally, we optimized our model *CheatSheetDPOaug* by applying 8-bit quantization on it. The quantization produced an interesting model that reduces the original model size from 3.23GB to 972.4MB which is more than 3 times size reduction. The difference in memory usage is significant and the tests did not show a relevant accuracy drop, indicating a working model. This can be explained by the fact that with 8-bit quantization, we lose some precision but still retain the most relevant part of the number representation, leading to a perfectly working model.

MCQA Accuracy	EPFL Dataset [%]
GPT-2	23.50
CheatSheetDPO	25.85
RAG1	28.85
RAG2	**29.70**
Quantized	23.93

Table 2: Accuracy across different step of fine-tuning and training

5 Analysis

SFT

While computing the metrics, we noticed a huge gap between BLEU and ROUGE scores and BERTScore. This can be explained by the fact that there are a lot of different ways to solve a math problem, name random variables or explain an algorithm. Therefore, naive metrics like BLEU and ROUGE fail to capture the completeness, corectness and overall clarity of the answer. On the other hand the very high BERTScore indicate that the contextual and semantical link between the answer and the generated sequence is maintained.

When evaluating code, n-gram similarity is less relevant because there are numerous ways to achieve the same functionality using different code implementations. We observed that the two models exhibited very similar performance, each achieving the best scores across different datasets and metrics in a balanced manner. This finding further supports the idea that the second half of the SFT training contributed less significantly to the model's learning.

DPO

We observed that the DPO-trained model on augmented dataset is less effective than the SFT version in determining the better answer from the preference pairs. This may be due to the fact that the EPFL dataset was collected by unpaid students with limited time, resulting in lower quality data compared to other datasets we used. Consequently, this can lead to lower accuracy, even for human evaluators. It is plausible that the DPO-trained model on the augmented dataset struggles to learn a behavior that satisfies all the datasets because of their disparities in difficulty. The use of additional, simple problems seem to somehow distill the information of EPFL problems.

Nevertheless, we argue that the model trained on the augmented dataset is more robust than the one trained exclusively on the lower-quality EPFL dataset as it achieves a greater margin (as illustrated as 4). Furthermore, the dataset labelled by paid individuals or collected by experienced people seem a lot more qualitative than the one collected by EPFL students. Given that the latter task involves specializing the network in multiple-choice question answering, we believe that *CheatSheetDPOaug* can achieve slightly better results overall, even if it is somewhat less effective in preference pair identification of the EPFL dataset.

MCQA generation

We observed that CheatSheet frequently predicts the letter 'A' if the prompt is not carefully designed. To achieve a well-distributed set of results, it is essential to use few-shot prompting. Without this approach, the probability of outputting any single letter, even when requested, is exceedingly low (typically on the order of 10^{-6}). We attribute this behavior to the fact that 'A' is the only letter that is also an English word, which likely explains its default selection over other letters.

Even after developing an appropriate prompt A.3, the performance of multiple-choice question answering (MCQA) remains suboptimal. This aligns with the findings of (Robinson et al., 2023) which report a Proportion of Prompted Answers (PPA) score close to 25% for GPT-2. This indicates that the model correctly selects the answer approximately one time out of four when the order of options is shuffled. Such extreme sensitivity to prompt structure appears to be an inherent limitation of using a pretrained GPT-2 model for MCQA tasks.

RAG

When performing RAG, we initially used three context dataset entries that we concatenated to the question. We obtained 25% accuracy, which is the same results as before performing RAG. This was mainly due to the fact that our model, based on `GPT2`, is limited to 1024 tokens and having three context entries to concatenate to the question often exceed this limit. Since the context comes before the question, this result in cutting it and therefore missing the question and obtaining a false answer. This architecture barrier that limit the input size of the prompt seemed hard to overcome and we could not figure a way to avoid it without truncating parts of plot.

Quantization

For quantization, as said, we experimented with several options. Firstly, we noticed no significant speed improvement in inference time despite setting the parameter `bnb_4bit_compute_dtype` to `torch.bfloat16`. This can be explained by the fact that we didn't perform extensive inference on our quantized models but instead tested them on a small test set. The reduction in operation size may nevertheless make a difference for extensive inference. On the other hand, we successfully reduced memory usage by loading our model in 4-bit quantization and in 8-bit quantization. The 4-bit quantization produced the smallest model, reducing our original model size from 3.23GB to 618.5MB, which is more than a five-fold reduction. Unfortunately, despite not producing a significant accuracy drop on the test set, we did not keep the 4-bit quantized model as a relevant result model because our manual observations showed that it consistently produced the same response letter for multiple-choice questions, making it irrelevant. This can be explained by the fact that reducing the model weights too much can cause the model to lose relevant information necessary for its proper functioning, rendering it obsolete. The 8-bit quantization, on the other hand, significantly reduces the model size while retaining the necessary information to maintain a working model with only a relatively small accuracy drop (1.92%).

6 Ethical considerations

As discussed in (Cotton et al., 2024), the use of AI for educational purpose can be double-sided. While it has the potential of offering many benefit to help engineering and science students, the creation of an expert academic question answering model also reflects many challenges such as academic integrity and plagiarism. Furthermore, the problem of scalability and accessibility has been a huge concern from the very beginning of AIs: while it was always possible to cheat, the accessibility of ChatBots has made it increasingly easier. This accessibility poses a temptation to resort to unethical means to attain higher grades, and may harm the educational system. As future engineers, we must consider such ethical aspects and try to account for them as much as possible. A remarkable tentative to limit the challenges risen by the use of ChatBots while fulfilling educational purpose is the work of (Kumar and Lan, 2024) that propose a model that guides students toward solving a problem by asking Socratic question independently without directly revealing the solutions.

Another aspect to consider is the fact that the model works only in English. We must adapt the model for other languages. It is important that our model is inclusive and accessible across different languages so that it benefits everybody and not just privileged people who had the opportunity to learn English during their education. To achieve this, we can use sequential few-shot transfer. Namely, we can further train our model on other datasets with high resource languages like French, Spanish or Chinese to learn a cross-lingual representation of words. These languages have rich corpora and well-developed natural language processing tools, making adaptation technically feasible. After this, we would fine-tune the model on small datasets from low-resource languages such as Urdu or Armenian. The small low-resource language datasets could be annotated by paid citizens of the respective countries.

While dealing with language, we must not forget the existing 300 different signed language, which are spoken by 72 millions people (Sig). Ensuring accessibility for users who communicate through signed language is crucial. This can be approached by incorporating signed language by integrating computer vision and gesture recognition technologies which enables the model to understand and respond to signed language inputs. This involves training the model with datasets of signed language gestures and their corresponding meanings, ensuring the model can accurately interpret and respond to diverse signing styles and collaborating with deaf communities to validate and improve the model's effectiveness.

7 Conclusion

In this project, we have demonstrated the potential of using a pretrained LM, specifically GPT2-large, for answering multiple-choice questions. Through our experimental design, we employed techniques such as SFT, DPO, RAG, quantization and prompt engineering to optimize the model's performance. Our study shows how each of these techniques improves the overall model.

GPT2 being a relatively small model, the results were not always perfect. Moreover, the alignment with the specific task of answering EPFL exam questions wasn't either, likely due to the low quality of the annotated preference pairs. However, we believe that with higher computational resources, bigger base model and better EPFL datasets, results could be significantly higher.

To conclude, this project provides a small step towards creating an effective AI-driven tool for assisting students in their learning journey at EPFL and beyond.

8 Team Contributions

Milestone 1

First milestone was mostly individual. We worked together to share our ideas and be sure that the selected papers covered a wide range of topics. We split the reading equally. The project proposal was the product of common agreements and discussions.

Milestone 2

ADAM BEN SLAMA

- First Model Idea
- Quantization reading
- DPO Dataset searching
- Report

MAHMOUD DOKMAK

- Dataset research and imports
- Data preprocessing
- Report

GARIK SAHAKYAN

- Code for `model_dpo.py`
- Data processing
- Code for training
- Report

BAPTISTE MAQUIGNAZ

- SFT and DPO training
- Code for `model_dpo.py`
- Metrics code and evaluation
- Report

Milestone 3

ADAM BEN SLAMA

- Quantization
- Report

MAHMOUD DOKMAK

- Quantization
- Report

GARIK SAHAKYAN

- RAG
- Report

BAPTISTE MAQUIGNAZ

- MCQA specialization
- Report

References

MS Windows NT kernel description. https://education.nationalgeographic.org/resource/sign-language/. Accessed: 2024-06-14.

NAMPDN AI. 2023. Tiny codes dataset. https://huggingface.co/datasets/nampdn-ai/tiny-codes. Version 1.0.

Argilla. 2023. Distilabel math preference dpo dataset. https://huggingface.co/datasets/argilla/distilabel-math-preference-dpo. Version 1.0.

ArtifactAI. 2023. Arxiv physics instruct tune 30k dataset. https://huggingface.co/datasets/ArtifactAI/arxiv-physics-instruct-tune-30k. Version 1.0.

Ayoubkirouane. 2023. arxiv-physics.

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.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Colin B. Clement, Matthew Bierbaum, Kevin P. O'Keeffe, and Alexander A. Alemi. 2019. On the use of arxiv as a dataset.

Debby RE Cotton, Peter A Cotton, and J Reuben Shipway. 2024. Chatting and cheating: Ensuring academic integrity in the era of chatgpt. *Innovations in education and teaching international*, 61(2):228–239.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*.

Jon Durbin. 2023. Py-dpo v0.1 dataset. https://huggingface.co/datasets/jondurbin/py-dpo-v0.1. Version 0.1.

GlaiveAI. 2023. Glaive code assistant dataset. https://huggingface.co/datasets/glaiveai/glaive-code-assistant. Version 1.0.

Pranav Hegde. 2023. Preference data math stack exchange dataset. https://huggingface.co/datasets/prhegde/preference-data-math-stack-exchange. Version 1.0.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

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

jilp00. 2023a. Youtoks-instruct-quantum-physics-ii.

jilp00. 2023b. Youtoks-instruct-quantum-physics-iii.

jilp00. 2023c. Youtoks-instruct-thermodynamics-of-materials.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Matthew Kenney. 2023. arxiv-math-instruct-50.

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

Nischal Ashok Kumar and Andrew Lan. 2024. Improving socratic question generation using data augmentation and preference optimization. *arXiv preprint arXiv:2403.00199*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

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

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2023. Gpt understands, too. *AI Open*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

MetaMath. 2023. Metamathqa dataset. https://huggingface.co/datasets/meta-math/MetaMathQA. Version 1.0.

Microsoft. 2023. Orca math word problems 200k dataset. https://huggingface.co/datasets/microsoft/orca-math-word-problems-200k. Version 1.0.

mjphayes. 2023. machine$_learning_questions$.

Mridul-Dixit. 2023. Machine-learning-qa-dataset.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

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

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

Rakesh Rajpurohit. 2023. Model quantization with hugging face transformers and bitsandbytes integration. *Medium*. Accessed: 2024-06-14.

Joshua Robinson, Christopher Michael Rytting, and David Wingate. 2023. Leveraging large language models for multiple choice question answering.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Shengyi Huang, Kashif Rasul, Alexander M. Rush, and Thomas Wolf. 2023. The alignment handbook. https://github.com/huggingface/alignment-handbook.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C. Schmidt. 2023. A prompt pattern catalog to enhance prompt engineering with chatgpt.

whiteOUO. 2023. Ladder-machine-learning-qa.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

A Appendix

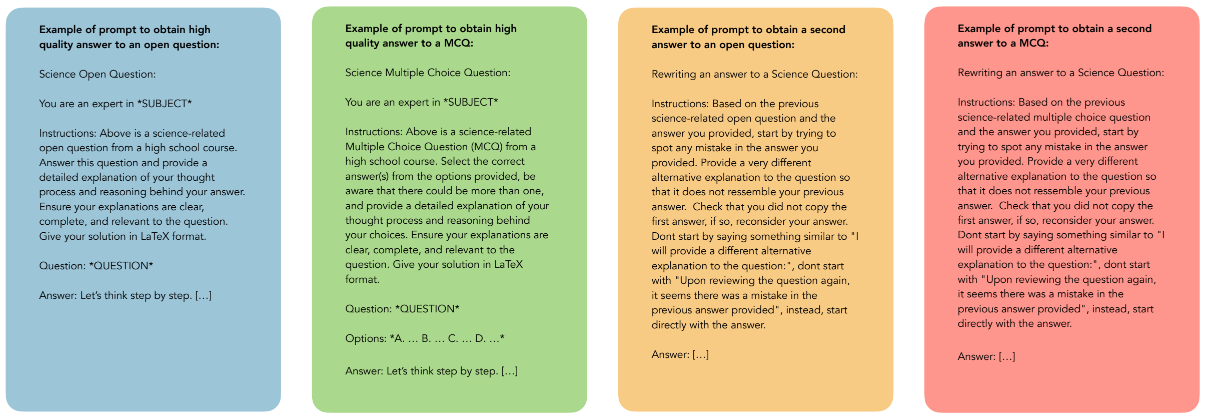
A.1 DPO Loss

The objective is to maximize the relative probability of the winning response compared to the losing response. The policy objective for the DPO training is

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

The loss function $\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}})$ is defined to optimize the policy π_θ against a reference policy π_{ref}. The expectation $E_{(x, y_w, y_l) \sim \mathcal{D}}$ is taken over a dataset $\mathcal{D}$ consisting of triples (x, y_w, y_l), where x is the input, y_w is the chosen (winning) response, and y_l is the rejected (losing) response. The term σ denotes the sigmoid function, and β is a scaling parameter. The probabilities $\pi_\theta(y \mid x)$ and $\pi_{\text{ref}}(y \mid x)$ represent the likelihoods of the responses y given the input x under the policy π_θ and the reference policy π_{ref}, respectively.

A.2 Prompt example to obtain preference pairs



Example of prompt to obtain high quality answer to an open question:

Science Open Question:

You are an expert in *SUBJECT*

Instructions: Above is a science-related open question from a high school course. Answer this question and provide a detailed explanation of your thought process and reasoning behind your answer. Ensure your explanations are clear, complete, and relevant to the question. Give your solution in LaTeX format.

Question: *QUESTION*

Answer: Let's think step by step. [...]

Example of prompt to obtain high quality answer to a MCQ:

Science Multiple Choice Question:

You are an expert in *SUBJECT*

Instructions: Above is a science-related Multiple Choice Question (MCQ) from a high school course. Select the correct answer(s) from the options provided, be aware that there could be more than one, and provide a detailed explanation of your thought process and reasoning behind your choices. Ensure your explanations are clear, complete, and relevant to the question. Give your solution in LaTeX format.

Question: *QUESTION*

Options: *A. ... B. ... C. ... D. ...*

Answer: Let's think step by step. [...]

Example of prompt to obtain a second answer to an open question:

Rewriting an answer to a Science Question:

Instructions: Based on the previous science-related open question and the answer you provided, start by trying to spot any mistake in the answer you provided. Provide a very different alternative explanation to the question so that it does not ressemble your previous answer. Check that you did not copy the first answer, if so, reconsider your answer. Dont start by saying something similar to "I will provide a different alternative explanation to the question:", dont start with "Upon reviewing the question again, it seems there was a mistake in the previous answer provided", instead, start directly with the answer.

Answer: [...]

Example of prompt to obtain a second answer to a MCQ:

Rewriting an answer to a Science Question:

Instructions: Based on the previous science-related multiple choice question and the answer you provided, start by trying to spot any mistake in the answer you provided. Provide a very different alternative explanation to the question so that it does not ressemble your previous answer. Check that you did not copy the first answer, if so, reconsider your answer. Dont start by saying something similar to "I will provide a different alternative explanation to the question:", dont start with "Upon reviewing the question again, it seems there was a mistake in the previous answer provided", instead, start directly with the answer.

Answer: [...]

A.3 Prompt for MCQA

Prompt used for MCQA:

Answer the following multiple-choice questions by selecting the correct letter (A, B, C, D).

Question: The solid-state structures of the principal allotropes of elemental boron are made up of which of the following structural units?
Options:
A. B12 icosahedra
B. B8 cubes
B6 octahedra
D. B4 tetrahedra
Answer: A

Question: Question: A machine learning problem involves four attributes plus a class. The attributes have 3, 2, 2, and 2 possible values each. The class has 3 possible values. How many maximum possible different examples are there?
Options:
A. 12
B. 24
C. 48
D. 72
Answer: D

Question: Of the following atoms, which has the lowest electron affinity?
Options:
A. F
B. Si
C. Ca
D. O
Answer: C

Question: The strongest base in liquid ammonia is?
Options:
A. NH3
B. NH2
C. NH4+
D. N2H4
Answer: B

Question: *QUESTION*
Options:
A. ...
B. ...
C. ...
D. ...
Answer: [...]