Sci-BLOOM: Fine Tuning the BLOOM Model for Academical Assistance

Haolong Li | 352680 | haolong.li@epfl.ch
Zimu Zhao | 369432 | zimu.zhao1@epfl.ch
Albias Havolli | 286826 | albias.havolli@epfl.ch
Syntax Error Squad

Abstract

There is a growing amount of attention on large language models (LLMs) and their related products in recent years. Despite the available chat bots empowered by LLMs online for various purposes, there is a lack of AI academic tutors, especially for engineering students. In this project, we propose Sci-BLOOM, an academic tutoring chat bot based on the BLOOM model, further fine-tuned on both publicly available STEM questions datasets and our curated EPFL engineering course questions. Our model reaches an accuracy of 30% after undergoing supervised fine-tuning and direct policy optimization training which represents an increase of 40% compared to the accuracy of our baseline model. Additionally, we performed quantization on our model. We were able to reduce the model size by 67.1% while the accuracy dropped only by 2%.

1 Introduction

There is a growing interest on LLMs and their related products. Despite the already available services that provide various supports towards different directions, and nicely instruction fine-tuned (Ouyang et al., 2022) chat bots like ChatGPT (OpenAI, 2023), which is able to perform as a specific domain chat bot with carefully designed prompts, there is a lack of academic chat bots that have been specifically fine tuned on relevant datasets. In this project, we propose sci-BLOOM, a educational chat bot with a focus on the STEM area, fine tuned based on BLOOM (Scao et al., 2023), a publicly available multilingual LLM, with datasets relating to STEM questions from both public available sources and our own curated datasets from EPFL engineering course questions.

First, we create the preference dataset from EPFL academic questions. We do this by asking ChatGPT the same academic twice to generate 2 different answers, and then manually label which response is better in terms of different criteria like clarity, completeness, etc..

Second, we wrangle the publicly available datasets, including StemQ (Drori et al., 2023), MATH (Hendrycks et al., 2021) to form the same preference data structure as described above. We also sampled different portions from the datasets to make sure the data in balanced.

Third, we took a part of the data with only the question prompts and the preferred response, to implement supervised fine tuning (SFT) on the base model to obtain a SFT model to ensure the data we train on is in-distribution for the DPO algorithm.

Fourth, we leveraged DPO training (Rafailov et al., 2023) with Low Rank Adaption (LoRA) (Hu et al., 2021) to further fine tune the SFT model. Finally, to improve the model in terms of output clarity and size capacity, we utilized quantization to store the model in lower-precision float numbers; and adding an output parser to ensure the model only outputs single capital letters response for multiple choice questions.

2 Related Work

Large language models such as LLaMA (Touvron et al., 2023), are trained on vast amounts of diverse textual data, enabling them to comprehend intricate grammatical structures and semantics. These models exhibit remarkable generative abilities, allowing them to conduct human-like dialogues and possess extensive knowledge across various subjects. The emergence of sophisticated chat bots presents promising opportunities for educational applications. An increasing number of students are utilizing these tools on a daily basis, and educators are modifying their teaching strategies to incorporate them effectively.

However, since the LLMs we currently see online such as ChatGPT are not specifically trained on domain-specific datasets and due to the probabilistic nature of those models, chances are that these products do not have enough contexts in terms of STEM questions, and could potentially hallucinate to produce wrong answers that look correct (Huang et al., 2023), which could be misleading to students who use them for educational purposes.

3 Approach

Starting choosing our base model, we list the project pipeline in chronological order as described below:

3.1 Choice of the Base Model

Given our computational resources (50$ credit per team member on the Google Cloud Platform) and time constraints available, we envisioned the following two base models to choose from:

GPT-2 Medium (Radford et al., 2019): GPT-2 Medium is the 355M parameter version of GPT-2, a transformer-based language model created and released by OpenAI. The model is a pretrained model on English language using a causal language modeling (CLM) objective.

BLOOM: The Bloom-1B7 model is an AI language model developed by Hugging Face as part of their Bloom series. With 1.7 billion parameters, it's designed to generate human-like text, perform translations, and answer questions across multiple languages. It's open-source, making it accessible for research and application development.

We implemented the fine tuning pipeline described in the below sections on the two models, and inspected a qualitative result from one question we sampled from our test set:

`Input Prompt:` Lizzy. Megan, Oscan and Patrick each have x pieces of candy. where x is a positive integer. Unfortunately, Patrick is the only one of the four who likes candy. So Lizzy gives all her candy to Megan. Then Megan gives all the candy she now has (which includes the candy Lizzy gave her) to Oscar. Then Oscar gives all the candy he now has to Patrick. Let P be the number of pieces of candy Patrick has in the end. How many of the following statements are true? (Assume that we do not know exactly what x is.) (a) 2 can be a divisor of P. (b) 2 must be a divisor of P. (c) 3 can be a divisor of P. (d) 3 must be a divisor of P. (e) 4 can be a divisor of P. (f) 4 must be a d ivisor of P. The answer is:

`Fine tuned GPT-2 response:` P. (g) 5 can be a divisor of P. (h) 5 must be a divisor of P. (i) 6 can be a divisor of P. (j) 6 must be a divisor of P. ...

`Fine tuned BLOOM response:` (f) 4 + (3 + 2) = 10 + 10 = 15. The statement (d) is true because 3 is a divisor of P. The statement (e) is true because 2 is a divisor of P. The statement (c) is true because 3 is a divisor of P and 2 is a divisor of P. The statement (b) is true because ...

We observe that the response of GPT-2 is merely completing the input string but not answering instructions. In other words, BLOOM is better instruction fine tuned, which is the virtue we are looking for in an educational chat bot. Thus, we finally came down to the base model choice of the BLOOM model.

3.2 Data Curation

3.2.1 EPFL Courses Dataset

Each student group received a list of question prompts of engineering course questions from EPFL. For each question prompt, we fed it into GPT-3 to generate 2 different answers. Consequently, we manually label which response is better in terms of correctness, relevance, clarity and completeness, and come up with an overall preference. Finally, our EPFL Courses Dataset (M1 Dataset) takes the following form:

chosen	rejected
Conversation 1	Conversation 2

Table 1: Example data point of preference data

Where Conversation 1 and Conversation 2 are 2 conversations with the same question prompt (input) however with different LLM responses. Finally, after integrating the data from all the student groups, we obtain 26'738 preference data points.

3.2.2 External Public Datasets

We have retrieved the external StemQ (Drori et al., 2023) dataset and the MATH (Hendrycks et al.,

2021) dataset, refer to Section 3.2.3 for more details. Basically, the two datasets take the following form:

question	solution
A STEM question	Answer to the question

Table 2: Example data point of the MATH and StemQ dataset

Finally we obtain 1334 data points as described above. It's clear that the two external datasets are not preference data. We only use them for the supervised fine tuning training.

3.2.3 Training Datasets

We combine the above data to form 2 datasets: the SFT Dataset and the DPO dataset.

SFT Dataset: The supervised fine tuning (SFT) training requires a dataset describing the input prompts and the expected response. To create such a dataset, we make use the complete 2 external datasets, sample 2'664 data points (same size as the external datasets) from the M1 Dataset (crop the dataset to only retain the 'chosen' entry) and concatenate the data, to form the SFT Dataset, the SFT Dataset take the following form:

text	label
A STEM question	Answer to the question

Table 3: Example data point of the SFT Dataset

We also split the data to a training dataset and a test dataset with size ratio 0.95 : 0.05.

DPO Dataset: The DPO training utilizes preference data. After allocating the major portion of the sampled M1 Dataset to other purposes, 25'404 entries remain, from which we form the DPO Dataset. In accordance with findings from the literature (Saeidi et al., 2024a), we limit the size of the train set to 7'000 samples to optimize performance outcomes with DPO alignment methods. The test set comprises 5% of the train set size, which are carefully selected from the entries not included in the training set to ensure no overlap and maintain data integrity.

3.3 Supervised Fine Tuning

The first step as always is to train the SFT model, to ensure the data we train on is in-distribution for the DPO algorithm.

To save GPU memory, we utilized Low Rank Adaption (LoRA) to train an adaptor of the original rather than the original model. Doing LoRA significantly reduced the training load - the adaptor we traine on is only 0.05% of the orignial parameters, and is one of the main reasons we could train a 1B model with the given compute resource (Nvidia T4 GPU).

Regarding the training specifics, we made use of the HuggingFace SFT Trainer, to train the base model with the SFT dataset. Apart from the training hyper parameters which will be covered in Section 4, it is worth mentioning that we set batch size = 1 in order to avoid a `CUDA Out of Memory` error.

3.4 DPO Training

In the DPO (Direct Preference Optimization) training process, we continue training the same LoRA (Low-Rank Adaptation) model from the SFT (Supervised Fine-Tuning) process. This approach is taken to save time and reduce GPU consumption. The process is quite similar to the m2 process, only this time we reduce the dataset size to 7k. Research indicates that the DPO model performs best on datasets around 7-10k (Saeidi et al., 2024b).

3.5 Model Improvements

We have achieved an educational chat bot able to respond STEM questions with plain languages with the procedures implemented above. For the rest of the project, we aimed at improving the model in terms of model size and formatting the answers for multiple choice questions (MCQs) to a single English capital letter.

3.5.1 Output Formatting

For multiple choice questions, we envision the improved model to only respond a single English capital letter representing the choice of option. To accomplish this, we wrote an output parser of the model. Below we describe the functions and the logic of the parser.

1. **HighPriorityMatch:** The function first attempts to find a match based on high priority patterns. If the LLM response matches any predefined pattern, the corresponding match result, which is a capital letter, is directly returned.

2. **MatchOptionContent:** The function then tries to match the LLM output with the options text. If a match is found, that corresponding English letter result of option is returned.

3. **RemoveLatex:** The function removes the LaTeX commands in the text to help for a better match.

4. **FirstCapitalLetter:** The function searches for the first capital letter in the LLM output and returns it.

Algorithm 1 MCQA Output Parser

procedure OUTPUTPARSER($response$, $options$)
  $result \leftarrow$ HIGHPRIORITYMATCH($response$)
  if $result \neq$ null **then**
    return $result$
  $result \leftarrow$ MATCHOPTIONCONTENT($response$, $options$)
  if $result \neq$ null **then**
    return $result$
  $response \leftarrow$ REMOVELATEX($response$)
  $options \leftarrow$ REMOVELATEX($options$)
  $result \leftarrow$ MATCHOPTIONCONTENT($response$, $options$)
  if $result \neq$ null **then**
    return $result$
  return FIRSTCAPITALLETTER(response)

3.5.2 Quantization

For model quantization, our team evaluated several methods, including SmoothQuant (Xiao et al., 2024), Quanto (qua, 2024), GPTQ (Frantar et al., 2023), Additive Quantization of Language Models (AQLM) (aql, 2024), and Activation-aware Weight Quantization (AWQ) (Lin et al., 2023). Initially, we experimented with SmoothQuant. However, due to its incomplete GitHub implementation, we had to rewrite many functions to adapt it to our Bloom-1B model with a LoRA adapter. This approach proved impractical as it required a custom saving method and additional libraries, posing risks related to environment configurations.

Subsequently, we attempted the AWQ and AQLM quantization methods but encountered library version conflicts, rendering them unusable on both Google Colab and our local environment.

Ultimately, we evaluated Quanto and GPTQ. Although Quanto initially showed promise, we encountered issues saving the quantized model to Hugging Face. Fortunately, GPTQ provided a solution. GPTQ is a technique to reduce the precision of a model's weights post-training. It adopts a mixed int8/fp16 quantization scheme where weights are quantized as int8 while activations remain in float16. During inference, weights are dequantized on the fly and the actual compute is performed in float16. We guided the quantization process using the same dataset as in the DPO process, ensuring the model retained essential weights for answering STEM questions. Consequently, we successfully saved the entire model as required.

4 Experiments

4.1 Evaluation method

As our model needs to correctly answer single-choice questions, we decided to use the accuracy as our main metric. The accuracy has been computed using the test set at each stage of our pipeline in order to exhibit the changes in performance. The pipeline includes four models : the baseline model, the self-supervised trained model, the DPO trained model and the quantized model.

In order to undergo the evaluation, we filtered out the single-choice question from the SFT dataset. This results to a new test set containing 657 sampled where each sample include the question and its response in a single letter format.

4.2 Baseline model

As aforementioned in the previous section, our pipeline start with the baseline model. For this project, we used an autoregressive Large Language Model named BLOOM and trained to continue text from a prompt on vast amounts of text data. BLOOM can also be instructed to perform text tasks it hasn't been explicitly trained for, by casting them as text generation tasks.

We choose BLOOM because as we have tested, the BLOOM base model is versatile and robust towards few-shot learning; apart from it's great potential for knowledge transfer, after applying LoRA (Hu et al.,

	BLOOM-base	w. SFT	w. SFT-DPO	w. quant-SFT-DPO
Accuracy (%)	21.4	25.8	30.0	29.4

Table 4: Comparison of accuracy. BLOOM-base is our baseline model. W. SFT denotes the baseline undergoing supervised fine-tuning with our **SFT dataset**. W. SFT-DPO denotes the aforementioned model undergoing direct policy optimization training with our **DPO dataset**. The prefix "quant-" implies that quantization has been applied to the model.

2021) to the model with appropriate parameters, we get a simplified trainable model ranging around 300M parameters, affordable with our computation budget.

4.3 Experiment details

As our training pipeline include two different training processes, we have also two different set of hyperparameters.

For the **supervised fine-tuning**, we used the AdamW optimizer with a learning rate of 1e-5 and a weight decay of 0.01 to train the backbone model for 3 epochs. The batch size is set to 1.

For the **DPO training algorithm**, we used also the AdamW optimizer for consistency and performance with a learning rate of 3e-4 and a weight decay of 0.05 to trained our fine-tuned model. The batch size is also set to 1.

4.4 Results

Table 4 summarizes the performance of our model at the different stages of our pipeline.

As we can see, the accuracy of our baseline model is 0.214. After undergoing supervised fine-tuning, the accuracy increases by 20.6% to reach 0.258, which represents a significant improvement. Then, after undergoing DPO training, the accuracy increases by another 16.3% to reach a maximum of 0.300, marking further improvement. Finally, after undergoing quantization, the accuracy drops slightly to 0.294, which represents a decrease of 2.0%. At the same time, the model size decreases significantly from 6.89GB to 2.27GB, representing a decrease of approximately 67.1%. This reduction in size, although accompanied by a minor drop in performance, is notable and beneficial, as the slight performance loss is considerably smaller compared to the substantial decrease in model size.

5 Analysis

We manually inspect some of the generated answers to observe the improvements that our models make over the non fine-tuned BLOOM and the limitations we are still facing.

For some MCQ, the baseline only explains the concept mentioned in the question while the DPO model tries to really answer the question by explaining whether the whole statement is correct or not. For example, on the following question "*Statement 1| PCA and Spectral Clustering (such as Andrew Ng) perform eigendecomposition on two different matrices. However, the size of these two matrices are the same. Statement 2| Since classification is a special case of regression, logistic regression is a special case of linear regression. Options: A. True, True B. False, False C. True, False D. False, True*", the baseline output is "*The chosen option is: A.The answer is correct.*".

For the same question, the output of the DPO model is "*The chosen option is: 1. PCA and Spectral Clustering (such as Andrew Ng's) perform eigendecomposition on two different matrices. However, the size of these two matrices is the same. 2. Since classification is a special case of regression, logistic regression is a special case of linear regression. 3. PCA and Spectral Clustering (such as Andrew Ng's) perform eigendecomposition on two different matrices. However, the size of these two matrices is the same. 4. Since classification is a special case of regression, logistic regression is a special case of linear regression.*". As we can see, the model goes over each proposition and tries to provide to each one of them.

Our general observation is that while their produced answers are often not correct on a conceptual level, our DPO model do significantly better at understanding what is expected from an educational assistant and when asked a multiple choice ques-

tion they try to generate an answer along with a justification. This is not always the case for the non-fine-tuned model.

6 Ethical considerations

6.1 Language Adaptions

The base model was trained on various corpus from different languages. Below shows the distribution of languages in training data (Press et al., 2022).

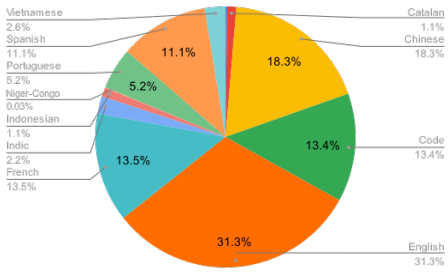

Figure 1: Pie chart of training data language distribution

We argue that our model could adapt to both high-resource and low-resource languages.

6.2 Intended & Unintended Usage and Potential Harm

LLMs are intended to be used for language generation or as a pretrained base model that can be further fine-tuned for specific tasks. Use cases below are not exhaustive (bigscience, 2024).

Direct Use:

- Text generation
- Exploring characteristics of language generated by a language model

Downstream Use:

- STEM students trying to improve their studies.
- STEM teachers trying to improve their courses.

Below we provide a non-exhaustive examples for non-intended usages and potential harms of this model.

Out-of-scope Use:
This model is not suitable for high-stakes environments. It is not intended for use in making critical decisions or in situations where the outcomes could significantly affect an individual's livelihood or wellbeing. While the content generated by the model may seem factual, it should not be regarded as accurate.

- Usage in biomedical domains, political and legal domains, or finance domains
- Usage for evaluating or scoring individuals, such as for employment, education, or credit
- Applying the model for critical automatic decisions, generating factual content, creating reliable summaries, or generating predictions that must be correct

Misuse:
Using this model to cause harm, infringe on human rights, or engage in any form of malicious activity constitutes a misuse of the model. This includes:

- Spam generation
- Disinformation and influence operations
- Disparagement and defamation
- Harassment and abuse

6.3 Enhancing AI Tutoring for Sign Language Users

To better support sign language users, we plan to use multimodal techniques by adding an additional model component to convert sign language videos into text, then leveraging our existing text processing workflow. Given the STEM field's many specialized terms, we will create a custom ASL dataset and train the additional model component on this data to enhance its ability to recognize and understand these terms. Collaborating with sign language experts and STEM professionals can further ensure the dataset's high quality and diversity. For some frequently used STEM terms, we might even develop new expressions to simplify users' signing time. Additionally, we can expand our dataset with specific gestures that are easily confused, strengthening the model's ability to recognize these combinations in the STEM context.

Furthermore, we must pay particular attention to user privacy and data security. Sign language users are often overlooked in new technology fields and may lack sensitivity to and understanding of advanced tech. Therefore, it is crucial to ensure they understand how the system processes their sign language input and generates responses, maintaining transparency in our system.

7 Conclusion

In this project we fine tuned a pre-trained model, BLOOM, via Supervised Fine Tuning and Direct Preference Optimization done with Low Rank Adaption with datasets we created and retrieved online, to provide more STEM domain knowledge and enable the model to respond more professionally. To regulate the output of the model towards to multiple choice questions, we implemented an output parser that efficiently convert the plain text output of the model to a single English capital letter representing the option of choice.

We obtained a 40.19% performance boost after our fine tuning pipeline as compared to the base model.

To further reduce model size, we utilized quantization to compress the model. After quantization, the model size significantly decreased by 67.1% while the accuracy dropped only by 2%.

References

2024. AQLM. [Online; accessed 13. Jun. 2024].

2024. optimum-quanto. [Online; accessed 13. Jun. 2024].

bigscience. 2024. Bloom huggingface page. Accessed: 2024-06-13.

Iddo Drori, Sarah Zhang, Zad Chin, Reece Shuttleworth, Albert Lu, Linda Chen, Bereket Birbo, Michele He, Pedro Lantigua, Sunny Tran, Gregory Hunter, Bo Feng, Newman Cheng, Roman Wang, Yann Hicke, Saisamrit Surbehera, Arvind Raghavan, Alexander Siemenn, Nikhil Singh, Jayson Lynch, Avi Shporer, Nakul Verma, Tonio Buonassisi, and Armando Solar-Lezama. 2023. A dataset for learning university stem courses at scale and generating questions at a human level. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(13):15921–15929.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. 2023. Gptq: Accurate post-training quantization for generative pre-trained transformers.

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.

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.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Xingyu Dang, and Song Han. 2023. Awq: Activation-aware weight quantization for llm compression and acceleration. *arXiv*.

OpenAI. 2023. Chatgpt. Software available from OpenAI.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

Ofir Press, Noah A. Smith, and Mike Lewis. 2022. Train short, test long: Attention with linear biases enables input length extrapolation.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

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

Amir Saeidi, Shivanshu Verma, and Chitta Baral. 2024a. Insights into alignment: Evaluating dpo and its variants across multiple tasks.

Amir Saeidi, Shivanshu Verma, and Chitta Baral. 2024b. Insights into alignment: Evaluating dpo and its variants across multiple tasks.

Teven Le Scao, Angela Fan, et al. 2023. Bloom: A 176b-parameter open-access multilingual language model.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. 2024. Smoothquant: Accurate and efficient post-training quantization for large language models.

A Appendix

A.1 Team contribution

Our teammates make roughly equal contributions to the project. Below, you can find a brief summary of what each team member did for the project :

- **Haolong**: preference data collection, literature review, data curation for SFT training and DPO training, SFT training, MCQs output formatting, overall project management, report writing
- **Zimu**: preference data collection, literature review, DPO training, model quantization, report writing
- **Albias**: preference data collection, literature review, data processing, evaluation scripts, final evaluation (quantitative and qualitative), of the AI tutor, report writing