# Progress Report: Milestone 2

André Santo | 376762 | andre.loureiroespiritosanto@epfl.ch
Pedro Chaparro | 374339 | pedro.chaparro@epfl.ch
Pierre Høgenhaug | 385070 | pierre.hogenhaug@epfl.ch@epfl.ch
**M**y **N**ice **L**ong **P**enguin

## 1 Introduction

Following our goal of creating an academic chatbot and having defined a plan, we continue our work by (1) collecting and processing data from different sources (2) vaguely training our elected baseline model with different datasets and hyperparameters to determine their ideal combination and (3) intensively train the model using Direct Preference Optimization (DPO). In this report, we discuss our progress, results and perspective on the forthcoming specialization.

## 2 Dataset

### 2.1 DPO Data Collection

We used a collection of 22k preference pairs of answers to EPFL questions, originating from other students, and collected 53k examples from 5 different datasets - **MMLU-STEM** (Lab, 2020), **TheoremQA** (Lab, 2023), **MathQA** (Chen et al., 2019), **arxiv-physics-instruct-tune-30k** (ArtifactAI, 2023), and **Open Platypus** (Lee et al., 2023). We used ChatGPT to generate the rejected answer and kept the original dataset answer as the chosen answer, assuming that ChatGPT would always generate an answer of lower quality. After setting aside data for testing, we created 4 unique datasets, of sizes **20k**, **30k**, **40k**, and **54k**, using different proportions of the gathered datasets, as visible in Figure 1. These generated datasets allowed us to partially train the same model and evaluate which proportion yielded the best results, using it to choose the final dataset used for DPO training. The selected datasets were optimal for our project due to their diverse sources, covering different aspects of STEM, providing a wide range of contexts and problem-solving examples.

### 2.2 SFT Data Collection

In milestone 3, we will specialize our model to answer MCQ. For that, we searched for datasets that were purely MCQ and ended up choosing **SciQ Dataset** (Welbl et al., 2017) and **Deepmind AQUA-RAT** (Ling et al., 2017). We chose these two specifically, for their size (a combined 107k MCQ) and they both came with a distinct column to support the correct answer which is something we will leverage on in the specialization of our model.

### 2.3 Data Processing

In the following section, we give a brief overview of the processing done to each dataset before combining them into the final DPO datasets.

We already had a general idea of the distribution of topics across the datasets. To gain more control for when we had to create combined datasets, we added a topic column to all of our entries. While **arxiv-physics-instruct-tune-30k** only consists of physics-related questions, **MMLU-STEM**, **Open-Platypus** and the EPFL data are more of a combination of topics. The **Open Platypus** came with labelled data sources for each entry, e.g. "PRM800K" or "SciBench" helping us sort out data sources irrelevant to the desired topics. To populate the topic column, we prompted ChatGPT, to identify which topic each entry was, giving it a list of topics to choose from. Since not all questions fall into the listed topics, and considering that ChatGPT might be unsure, we also considered the option "Unknown". We manually processed all entries labelled "Unknown", to either assign it a topic or to drop the entry.

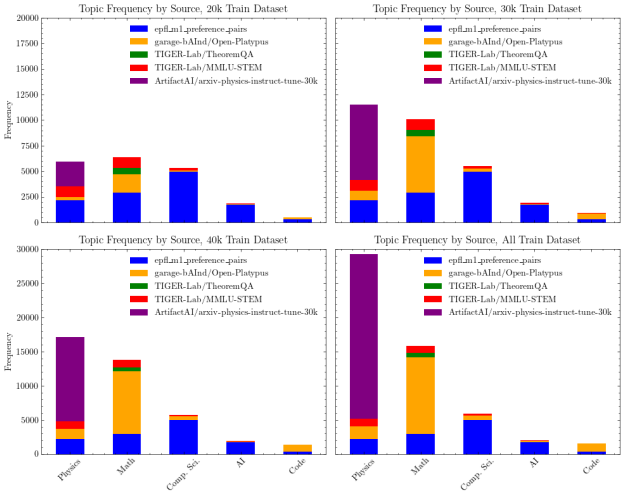



Figure 1: Train Dataset Source & Topic Split

Across all the datasets we gathered online, we identified MCQ without explanations to their answers. For these questions, we prompted ChatGPT to take the correct answer and explain why that answer is correct and, before creating the combined datasets, we ensured that all datasets had identical column names, facilitating the merging.

Finally, for each dataset we created a 80/20 train/test split. For the EPFL preference pair dataset, we ensured that for each unique question, all preference pairs had to be in either the train or the test split and never in both.

## 3 Model

### 3.1 Model Architecture and pre-training

Phi-2 (Gunasekar et al., 2023) is an autoregressive model from Microsoft with 2.7B parameters and being known for

its good performance in reasoning, surpassing many larger models. We opted to use a supervised fine-tuned (*SFT*) version from HuggingFace. It has been *SFT*'d and uses the ChatML template. Some of the datasets used for *SFT* are of interest for us, such as Synthia-v1.3 and Verified-Camel, so we kept using the same template in our training. As it had been finetuned, we can immediately apply *DPO*.

### 3.2 Training Loss

As we cannot prove the quality of the data provided by the students, nor the quality of the data collected online, we opted to use a different type of loss than *sigmoid*. We chose to use **Robust DPO Loss** (Chowdhury et al., 2024) which is more robust to random preference flips.

## 4 Preliminary training results

Given the produced datasets presented in Section 2.1, to elect the best dataset, we trained the model with 1350 samples of each dataset for three epochs and obtained the results presented in Table 1. **Best Test Loss** refers to the lowest loss recorded during the evaluation phase of training (*eval/loss* in Tensorboard) of the three epochs and **Evaluation Loss** is the computed a posteriori loss of the evaluation dataset, a fixed set of 5000 samples extracted from the 54k dataset i.e. inference with the trained models was made and the loss between the generated and expected answers was calculated.

| Dataset Size | Best Test Loss | Eval. Loss |
|---|---|---|
| 20k | 0.43 | 2.4830 |
| 30k | 0.4113 | 2.4313 |
| 40k | 0.3409 | 2.4333 |
| 54k | 0.276 | 2.5349 |

Table 1: Datasets Comparison

We believe the results were not ideal, as we observed both (1) increasing test loss and (2) signs of overfitting on Tensorboard for some datasets. We theorize some explanations: the amount of training data could have been insufficient; A perhaps too high learning rate, 1e-4, was used, possibly making the model converge to local minima; For the best test loss, it might not be directly comparable between sizes, as different test datasets were used, each being created from the dataset being measured.

Despite that, we considered our main dataset the **40k dataset**: both the 30k and 40k had the lowest evaluation losses and, as such, were good candidates, so the lower best test loss of the 40k dataset was the deciding factor.

Having decided the dataset, we proceeded with hyperparameter finetuning. We considered three parameters: the **Learning Rate**, the **Beta** parameter of DPO, and the **Label Smoothing** parameter of Robust DPO. For each, we tested four different values and, using 3000 samples of the 40k dataset, trained a total of 16 versions of the model for 3 epochs each and observed the lowest (i.e. best) test loss of the three epochs. The results can be observed in Table 2.

The elected parameters are highlighted in bold on the table. For the learning rate, we chose 1e-5 as, looking at Tensorboard, both the *eval/loss* and the *train/loss* go down in a steady and expected manner, unlike 3e-5 and 5e-5. For beta, we went with the lowest value and for label smoothing

| L.r. | B.T. Loss | Beta | B.T. Loss | L. Sm. | B.T. Loss |
|---|---|---|---|---|---|
| 5e-6 | 0.2121 | 0.05 | -12.41 | 0 | 0.2848 |
| **1e-5** | **0.0624** | 0.1 | -7.322 | 0.05 | 0.0537 |
| 3e-5 | -7.786 | **0.3** | **-59.12** | **0.1** | **-9.94** |
| 5e-5 | -0.0819 | 0.5 | -4.293 | 0.25 | -241.5 |

Table 2: Hyperparameters Comparison

| Model | Chkpt | Acc |
|---|---|---|
| 40k default | final | 0.744 |
| 40k default | 1000 | 0.6306 |
| 40k beta 0.1 | 2500 | 0.743 |
| 40k beta 0.1 | 1000 | 0.6136 |
| 20k default | final | 0.6292 |

Table 3: Model, checkpoints and accuracy on policy reward

we observed the same behavior of the learning rate and so chose a sub-optimal value.

Nonetheless, we also trained two alternative versions: one with beta = 0.1, and another using the 20k dataset. The resulting accuracies, for different checkpoints, can be observed on Table 3.

To evaluate the performance, we only considered the accuracy, present in the table, as well as the evaluation loss, observed in Tensorboard but not shown in this report. In both metrics, **40k default** performed best and so we elected it to be used for training in the last milestone.

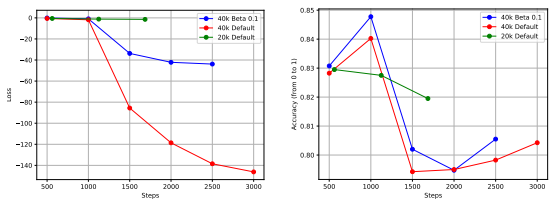
For now, we only considered those metrics as they are the most meaningful to the goal and to what the model will be evaluation on. Nonetheless, for milestone 3, we plan to further measure its performance and compare it to others, using metrics such as BLEU, ROUGE, BERTScore, and human and automatic evaluation (using ChatGPT's API).

Having chosen the configuration, we plotted the evolution of the evaluation loss and accuracy, shown in Figure 2.

## 5 Quantization Specialization

For the specialization of our model, we decided to follow the quantization path. Hence, all the baseline models we considered, and the one we are using, were all of enough precision to allow us to apply quantization.

Currently, we plan to quantize the final model, using libraries like bitsandbytes (Face), with different precisions and compare the size and performance differences across models. If time allows, we will additionally perform Quantization Aware Training (QAT) (Flow) and compare the results, but this would require additional data collection.




(a) Loss (b) Reward Accuracy

Figure 2: Evaluation Loss and Accuracy using 10% of `combined_40k_train.jsonl`

## References

ArtifactAI. 2023. Arxic physics instruct tune 30k model card.

Wenhu Chen, Ming Yin, Max Ku, Pan Lu, Yixin Wan, Xueguang Ma, Jianyu Xu, Xinyi Wang, and Tony Xia. 2019. Mathqa model card.

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

Hugging Face. bitsandbytes library explanation.

Tensor Flow. Quantization aware training.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Harkirat Singh Behl, Xin Wang, Sébastien Bubeck, Ronen Eldan, Adam Tauman Kalai, Yin Tat Lee, and Yuanzhi Li. 2023. Textbooks are all you need.

TIGER Lab. 2020. Mmlu-stem model card.

TIGER Lab. 2023. Theoremqa model card.

Ariel N. Lee, Cole J. Hunter, and Nataniel Ruiz. 2023. Open platypus model card.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. *arXiv preprint arXiv:1705.04146*.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions. *CoRR*, abs/1707.06209.