# Progress Report: ProbLLMs

Arvind S. Menon | 354584 | arvind.menon@epfl.ch
Lars C.P.M. Quaedvlieg | 352130 | lars.quaedvlieg@epfl.ch
Lazar Milikic | 353622 | lazar.milikic@epfl.ch
Somesh Mehra | 353628 | somesh.mehra@epfl.ch
GradientVanishers

## 1 Introduction

In this project, we aim to train an AI assistant to help students with multiple-choice questions from EPFL courses. This progress report describes the generation of additional datasets, current training approaches to align our model, and further steps for future model specializations.

## 2 Dataset

We have constructed a dataset of preference pairs for fine-tuning models, utilizing multiple sources.
**MNLP Students-Annotated Dataset:** This dataset initially comprised approximately 22,000 preference pairs samples, gathered from interactions with ChatGPT. These interactions involved answering various questions related to Computer Science, Machine Learning, and Physics courses at EPFL. Students annotated the responses based on several criteria, including correctness, relevance, clarity, and completeness. For our project, which aims to enhance LLMs' accuracy in responding to EPFL course-related queries, we specifically filtered out any preference pairs where both the chosen and rejected answers were deemed incorrect.

**Computer Science Theory QA Dataset:** This dataset, publicly available on Kaggle (Mateen, 2023), comprises question-answer pairs covering 150 subtopics across various Computer Science and Machine Learning disciplines. To construct a preference pair dataset, we adopt the methodology proposed by (Huang et al., 2023), which involves transforming correct answers into rejected ones. Specifically, we employ three ChatGPT-driven prompts to corrupt the answers: *REMOVE* essential content, *SUBSTITUTE* parts of the answer with incorrect information, or *INSERT* irrelevant details. For each transformation, two out of the three corruption techniques are randomly applied to modify the original correct answer into a rejected one. Detailed descriptions of these corruption prompts are available in Appendix A.

**Camel-AI Math & Physics Datasets:** These two datasets contain over 70,000 examples from various math and physics questions (Li et al., 2023). Although the datasets are large, the data was synthetically generated by GPT-4 and may contain inaccuracies as noted by the authors. Each dataset consists of question-answer pairs, allowing us to apply the same corruption techniques described earlier to generate rejected answers. Due to budget constraints with ChatGPT, we sample approximately 2,500 samples from the Camel-AI Math dataset and about 3,000 from the Camel-AI Physics dataset. We allocate more samples to the physics dataset, as we primarily work with LLMs that are already tuned on math datasets, see Section 3.

**Stack Exchange Preferences Dataset:** This dataset comprises preference data from over 10 million questions sourced from various Stack Exchange forums, including Stack Overflow, Mathematics Stack Exchange, Physics Stack Exchange, Computer Science Stack Exchange, etc. (Lambert et al., 2023). Each question is associated with multiple answers, receiving a preference score based on user upvotes. We selectively compile preference pairs by choosing a "chosen answer" that not only has the highest preference score among the answers but also possesses a minimum score of 10. Simultaneously, we identify a "rejected answer" with a score between 3 and 5. This selection criterion ensures a high-quality "chosen answer" while the "rejected answer" is considered adequate, yet maintaining a significant quality gap between them. We exclude examples that do not meet these conditions. Through this method, we have collected approximately 42,000 samples across various relevant domains.

Additionally, we have collated various multiple choice question answering (MCQA) datasets to evaluate the accuracy of our model for generating correct answers.

**ARC:** This dataset contains grade-school level multiple-choice science questions. The dataset is partitioned into an Easy and Challenge set with 5,196 and 2,590 questions respectively (Clark et al., 2018).

**MMLU:** This dataset contains multiple-choice questions from various topics (Hendrycks et al., 2021a). We only keep a subset of relevant science-related topics (Appendix D), leaving us with 4219 questions.

**GSM8K:** This dataset contains 8792 grade school math questions (Cobbe et al., 2021), with each answer containing an explanation and a final numeric answer. To convert it into a MCQA dataset, we extract the final answer and synthetically generate three additional options by adding random values between -10 to 10 to the actual answer. The final options are also randomly shuffled.

**MATH:** This dataset contains 12,500 challenging competition mathematics problems, each with step by step solutions that contain a free-form answer inside a '\boxed{}' element (Hendrycks et al., 2021b). Similar to GSM8K, we convert this into an MCQA dataset by synthetically generating three additional answer options. Since the answers are not always a single number (for example they can contain fractions, square roots etc.), to generate reasonable options, we first parse all the numbers from the answer, and randomly select one number to add a value between -5 to 5.

## 3 Model

We opt for the DeepSeekMath-RL (Shao et al., 2024) model, which is a 7 billion-parameter model. They obtain this model by mathematical instruction fine-tuning and further Group Relative Policy Optimization (GRPO) (Shao et al., 2024) on the DeepSeekMath model. The DeepSeekMath model continues pretraining DeepSeek-Coder-Base-v1.5 (Guo et al., 2024) 7B with 120B math-related tokens sourced from Common Crawl, together with natural language and code data. We describe the architecture of this model in Table 3. Finally, the DeepSeek-Coder-Base-v1.5 was pretrained with 2.0T general text tokens and additional code tokens.

This model performs competitively with LLama-3 70B (Touvron et al., 2023) on existing mathematics benchmarks, as discussed in the project proposal.

We will use Direct Preference Optimization (DPO) (Rafailov et al., 2024) with Low-Rank Adaptation (LoRA) (Hu et al., 2021) for parameter-efficient alignment to the datasets described in Section 2.

## 4 Preliminary Training Results

### 4.1 Experimental Setup

We train two models on one NVIDIA A100 (80GB) GPU, which have the same hyperparameter configuration but a different data distribution:

- Student-only model (7B): This model is trained on a uniformly random 80%/20% train and test split of only the MNLP students preference dataset. We use this as a baseline to see if the additional collected datasets from Section 2 provide any additional value for reward accuracies.

- ProbLLM-7B: This model is trained with the same training and test split of the MNLP students preference datasets as the Student-only model. However, for this model, we also include all additional collected data to the training set.

Due to the costly nature of fine-tuning a 7B model with our limited compute, we perform a limited search over the hyperparameters, where we manually tweaked certain values such as the learning rate schedule, LoRA configuration, and more. The extensive list of final hyperparameters that were used for both models can be found in Section C.

In this milestone, we use the reward accuracy of the previously described evaluation dataset as a performance measure of the models. We additionally provide a few sample generations to qualitatively evaluate the performances. In future milestones, we will utilize the additional evaluation datasets described in Section 2, but we did not have sufficient time to benchmark with this in this milestone.

### 4.2 Results

| Model | Evaluation Loss | Train Reward Acc. | Test Reward Acc. |
|---|---|---|---|
| Student-only | 0.8811 | 0.9969 | 0.5687 |
| ProbLLM-7B | 0.6356 | 0.9125 | 0.6164 |

Table 1: Certain metrics at the end of training for both models. "Acc" refers to accuracy.

From the evaluation loss plots in Appendix E, one can observe that the student-only model ends up heavily overfitting on the training dataset. However, ProbLLM-7B does not attain to such a level of overfitting. Instead, when looking at the evaluation reward accuracy, it steadily increases instead of constantly decreasing like the Student-only model. This can also be observed in Table 1.

## 5 Retrieval Augmented Generation

As specified in our proposal, we plan to leverage retrieval augmented generation (RAG) to further improve our model. For this, we will simply incorporate the additional retrieved context into our prompts when calling our trained model, thus no further training will be required. Relevant context will be retrieved using a search of our vector index, using the embeddings of the question. We will populate our vector index with relevant course notes and textbooks, which we have started collating.

For evaluation, we will perform an ablation study on the accuracy of MCQA, to understand the contributions of RAG and our finetuning compared to the base model we used. Additionally, we will sample a subset of questions from the student annotated dataset to generate extended responses (with reasoning) using our model with and without RAG, and use GPT4 as a judge (Zheng et al., 2024) to evaluate if RAG improves the quality of generations.

## 6 Quantization

The model we have fine-tuned uses 16-bit floating point numbers. To significantly compress our model for lower memory requirements and faster inference, we plan to test 4- and 8-bit quantization. We will evaluate these models on their accuracy on the MCQA datasets outlined in Section 2, to judge which precision offers the best compression-accuracy tradeoff. We will compare the performance of the quantized models against the original model, as well as various baselines. For the baselines, since we are quantizing a very large 7B model, we will finetune smaller models which have a similar size to our quantized models, to judge whether finetuning and quantizing a large model was more effective than simply finetuning a smaller model.

## References

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

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y Wu, YK Li, et al. 2024. Deepseek-coder: When the large language model meets programming–the rise of code intelligence. *arXiv preprint arXiv:2401.14196*.

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

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

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. *arXiv preprint arXiv:2106.09685*.

Shijia Huang, Jianqiao Zhao, Yanyang Li, and Liwei Wang. 2023. Learning preference model for LLMs via automatic preference data generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9187–9199, Singapore. Association for Computational Linguistics.

Nathan Lambert, Lewis Tunstall, Nazneen Rajani, and Tristan Thrush. 2023. Huggingface h4 stack exchange preference dataset.

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.

Mujtaba Mateen. 2023. Computer science theory qa dataset.

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.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models.

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*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2024. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36.

## A Corruption promputs

We provide the specific prompts used to corrupt answers for generating the preference pairs as described in the (Huang et al., 2023). These prompts enable the transformation from correct to rejected answers through deletion, substitution, and insertion of content.

| Corruption Type | Example |
|---|---|
| **Deletion** | You are an adversarial actor trying to corrupt the correct answers. Your goal is to REMOVE some answer content that is useful to the question. |
| **Substitution** | You are an adversarial actor trying to corrupt the correct answers. Your goal is to EDIT some parts of the answer content to make it INACCURATE for the question. |
| **Insertion** | You are an adversarial actor trying to corrupt the correct answers. Your goal is to ADD some answer content irrelevant to the question into the answer. |

Table 2: Detailed Examples of Corruption Prompts

## B Model Architecture

| **Params** | $n_{layers}$ | $d_{model}$ | $n_{heads}$ | $n_{kv\_heads}$ | **Context Length** |
|---|---|---|---|---|---|
| 7B | 30 | 4096 | 32 | 32 | 4096 |

Table 3: Detailed specs of DeepSeek LLM family of models.

## C Training Hyperparameters

| Parameter | Value |
|---|---|
| Seed | 0 |
| Precision | bfloat16 |
| Optimizer | paged_adamw_32bit |
| Top_k | 50 |
| Top_p | 1 |
| Steps | 2000 |
| Epochs | 1.533 |
| Adam_beta1 | 0.9 |
| Adam_beta2 | 0.999 |
| Eval_steps | 100 |
| Hidden_act | silu |
| Max_generation_length | 1024 |
| Vocab_size | 102400 |
| Adam_epsilon | 1e-8 |
| Learning_rate | 0.0005 |
| Learning_rate_schedule | cosine |
| Warmup_ratio | 0 |
| Warmup_steps | 100 |
| Weight_decay | 0.05 |
| Max_grad_norm | 1 |
| Lora_beta | 0.05 |

| Parameter | Value |
|---|---|
| Lora_alpha | 16 |
| Lora_r | 8 |
| Lora_dropout | 0.05 |
| Batch_size | 8 |

## D MMLU Topics

We retain questions from the following subjects in the MMLU dataset, organised by topic and with their respective question counts:

- **Math:**
  - Abstract Algebra (116)
  - Elementary Mathematics (424)
  - College Mathematics (116)
  - Formal Logic (145)
  - High School Mathematics (304)
  - High School Statistics (244)
- **Natural Sciences:**
  - Anatomy (154)
  - Astronomy (173)
  - College Biology (165)
  - College Chemistry (113)
  - Conceptual Physics (266)
  - College Physics (118)
  - High School Biology (347)
  - High School Chemistry (230)
  - High School Physics (173)
  - Virology (189)
  - Medical Genetics (116)
- **Computer Science:**
  - College Computer Science (116)
  - High School Computer Science (114)
  - Machine Learning (128)
  - Computer Security (116)
- **Engineering:**
  - Electrical Engineering (166)
- **Other:**
  - Logical Fallacies (186)

## E Additional Graphs

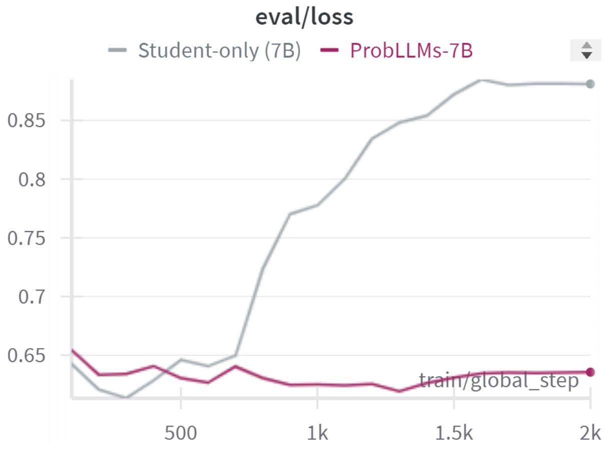

Figure 1: Evaluation loss

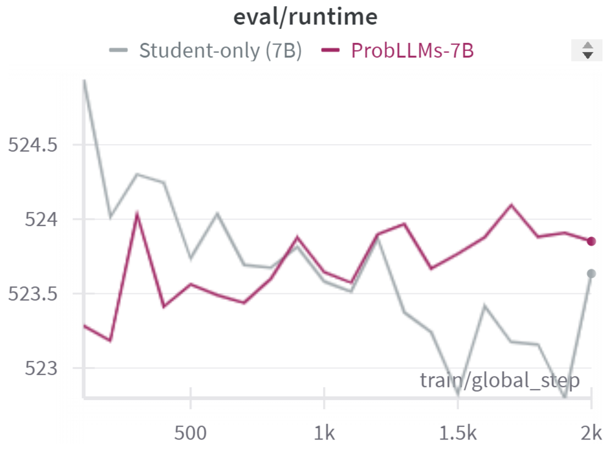

Figure 2: Evaluation runtime

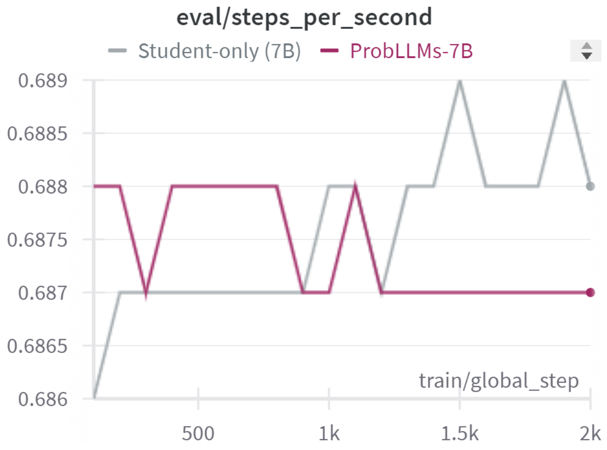

Figure 3: Evaluation steps per second

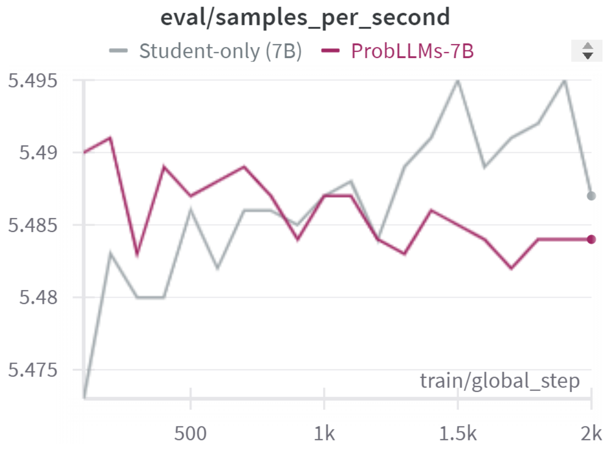

Figure 4: Evaluation samples per second

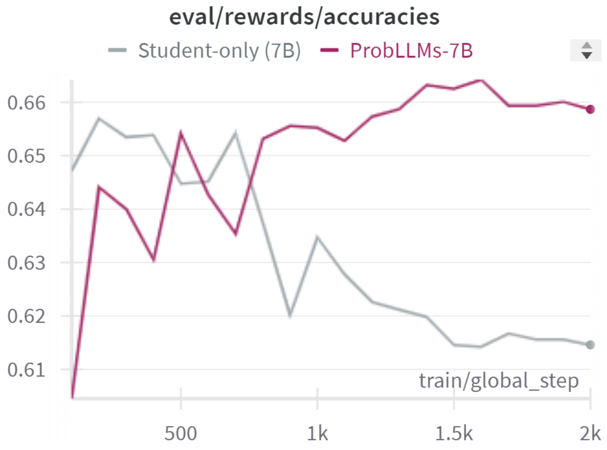

Figure 5: Evaluation reward accuracy

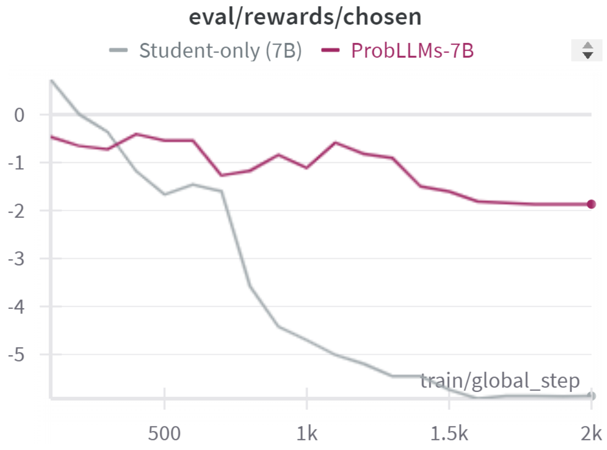

Figure 6: Evaluation chosen samples reward

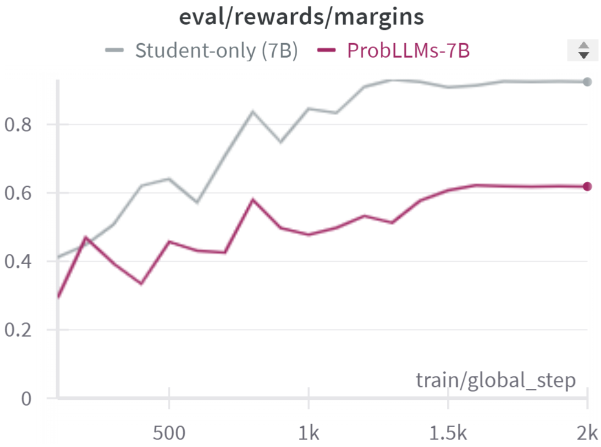

Figure 7: Evaluation reward margin

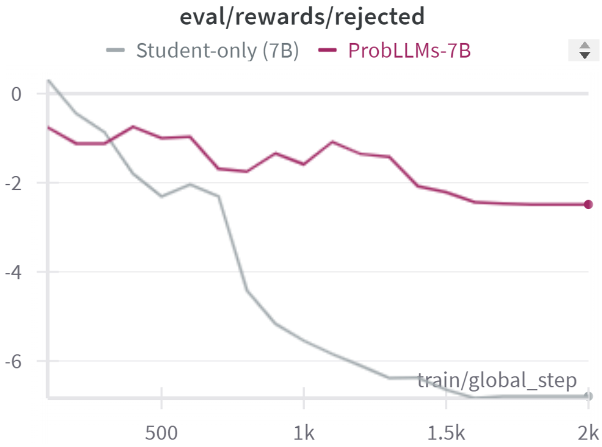

Figure 8: Evaluation rejected samples reward

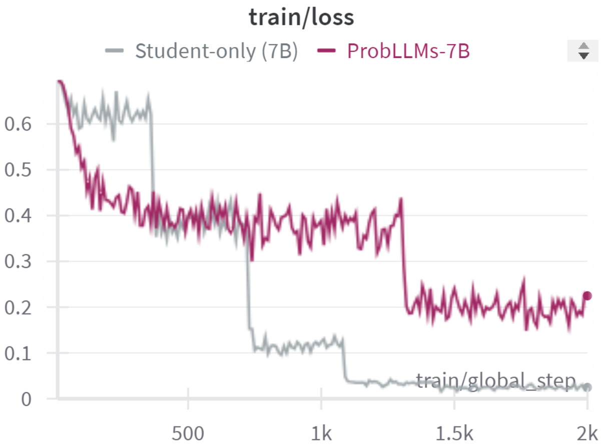

Figure 9: Training loss

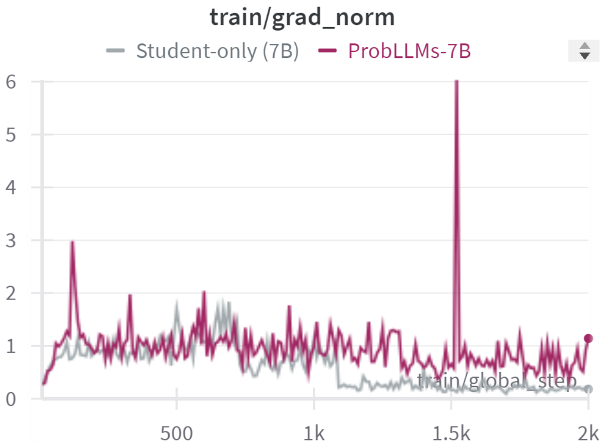

Figure 10: Training gradient norms

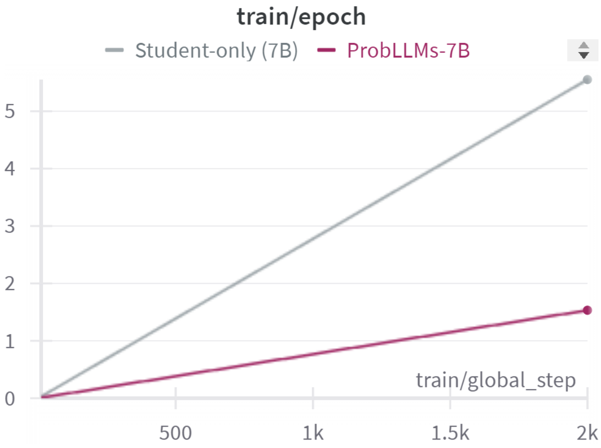

Figure 11: Training epoch count

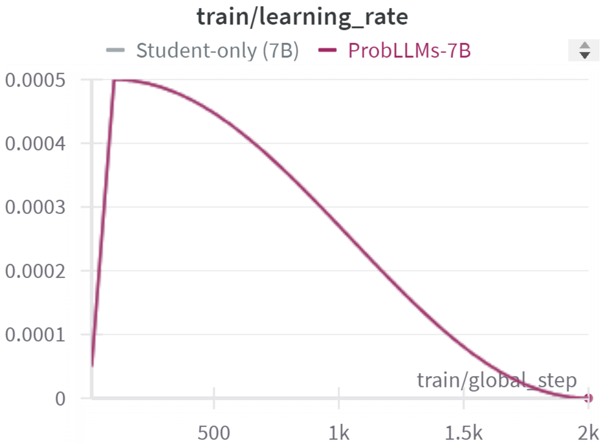

Figure 12: Training learning rate

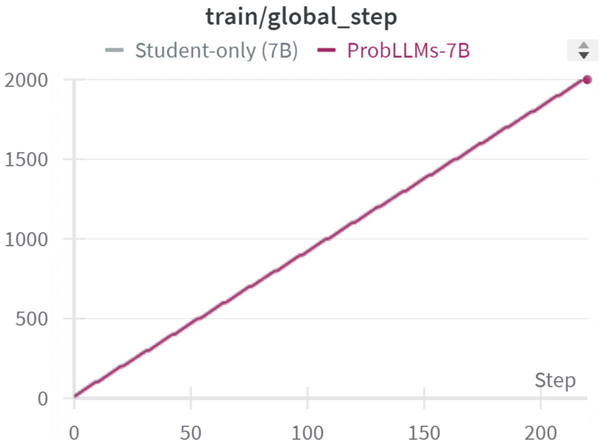

Figure 13: Training global step

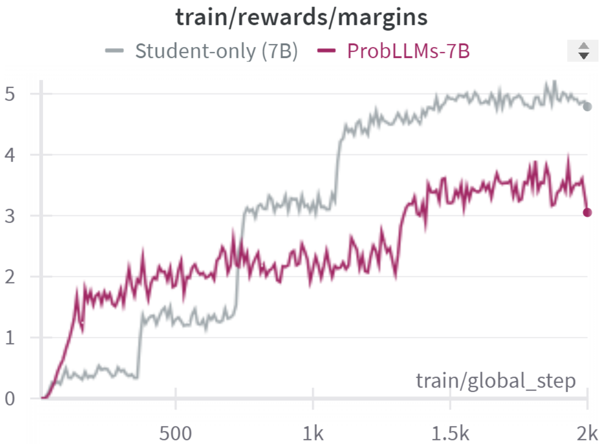

Figure 14: Training reward margins

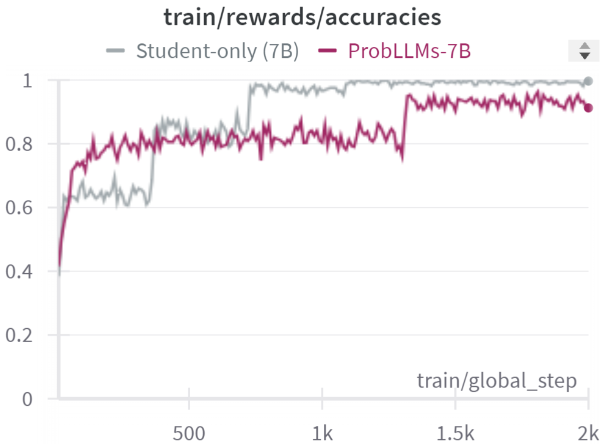

Figure 15: Training reward accuracies

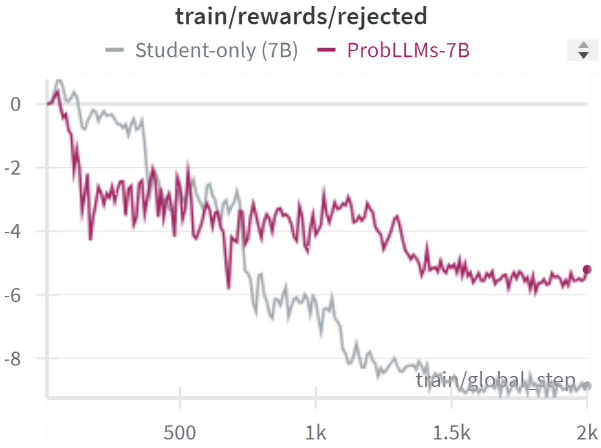

Figure 16: Training reward rejected sample

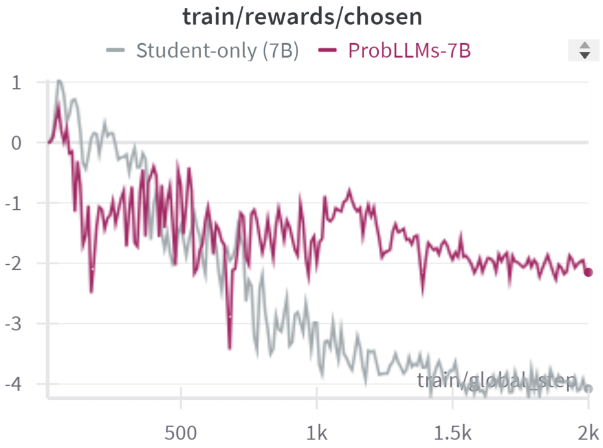

Figure 17: Training reward chosen sample