FinesHerbesMCQA: Enhancing QWEN3-0.6B Base for STEM subjects Multiple Choice Question Answering.

Valentin Perret | 327718 | valentin.perret@epfl.ch
Jehan Piaget | 325390 | jehan.piaget@epfl.ch
Lucie Huamani-Cantrelle | 297795 | lucie.huamani-cantrelle@epfl.ch
Gustave Lapierre | 326066 | gustave.lapierre@epfl.ch
AI et fines herbes

Abstract

In this project, we introduce FinesHerbesMCQA, a model fine-tuned from Qwen3-0.6B-Base for Multiple Choice Question Answering (MCQA) on STEM subjects. The model leverages Direct Preference Optimization (DPO) for improved alignment with human preferences. We derive two variants: a quantized version with 8-bit weights and activations, and a Retrieval-Augmented Generation (RAG) model. Our approach integrates Stepwise DPO (sDPO), Supervised Fine-Tuning (SFT), and post-training quantization (PTQ) of both weights and activations. We show that model performance is highly sensitive to the task formulation and the choice of objective function.

1 Introduction

Recent advances in Large Language Models such as GPT-2 (Veyseh et al., 2021) have made it possible to solve a large variety of tasks. In this context, students use it in their advanced studies, to the extent that it was suggested that LLM could obtain an engineering degree (Borges et al., 2024).

Within the scope of the Modern Natural Language Programming class, we learned about various methods of fine-tuning an LLM to solve a task. Therefore the goal of this project is to fine-tune such a model so that it becomes our teaching assistant. There exist different ways of evaluating model generation, however it remains a challenge, (Gao et al., 2025). The simplest way for the project duration is to evaluate the model on MCQAs about advanced STEM subjects.

The fine-tuned model is the Qwen3-0.6B-Base model (Team, 2025), which is aligned with human-like data via DPO training. Furthermore, several alternatives of the model are developed: a fine-tuned version, for smaller storage purposes a quantized version, and finally, to enhance its knowledge we developed it with a RAG model as well.

2 Approach

We fine-tune Qwen3-0.6B-Base, being a model with only 0.6 billion parameters. We decided to fine-tune the whole model entirely for the DPO and MCQA training.

With regard to the **Reward model approach**, we adopt the sDPO approach (Lai et al., 2024), an extension of DPO for aligning LLMs with human preferences. Instead of training with all the data at once, it uses a curriculum learning paradigm, where training data is partitioned into subsets of increasing difficulty. This allows the model to first align on easy preference pairs and progressively handle more subtle differentiations. In our setup, the preference dataset is divided into $T = 3$ partitions: *easy*, *medium*, and *hard*, based on an estimated difficulty ranking. Data coming from UltraFeedback dataset (GPT-4 annotated) makes our setting partially aligned with the Distilled DPO (DDPO) framework (Fisch et al., 2025), where preferences distilled from stronger models are used to supervise smaller models.

The **MCQA task** is defined as either a single token prediction or a multi token prediction, either the label or the label with the choice proposition is predicted. We want to experiment with both methods. Furthermore, we want to increase the probability of the correct token. For this, we base our approach on one main method.

We implemented, **Supervised Fine Tuning** (SFT), a method consisting of aligning the prediction of a model with the completion of a prompt. It minimizes the negative log likelihood of the next token given the previous prompt. If we define the loss for the token in position t, $\mathcal{L}_t$, we have the relation

$$\mathcal{L}_t = -\log P(y_t^* \mid \{y_{<t}^*\})$$

where y^* is the ground truth token. The model increases the probability of the gold token at instant t.

We believe that in our case it would learn answers to questions. It pays attention to the previous tokens; therefore, by seeing many different questions and subjects, we expect the model to gain some knowledge. We also want to test a method with CrossEntropyLoss, the probability is computed on the four options, and thus penalizes if the correct option is not probable enough.

Furthermore, we provide additional context using a RAG framework. We will create a relevant STEM-focused dataset to serve as the retrieval corpus. We will select an appropriate encoder and similarity function to embed both questions and documents and retrieve the most relevant context passages.

Finally, we quantize the final MCQA model using the `llmcompressor` library, applying a single-pass `oneshot` PTQ.

3 Experiments

3.1 Data

All datasets, split in test and validation set, were preprocessed to focus on STEM subjects to ensure the model is trained on meaningful and in-domain examples.

3.1.1 DPO data

To align our model with human preferences, we combine three datasets: UltraFeedback Binarized cleaned, Epfl preference pairs annotated by students, and StackExchange.

In contrast with the work proposed by sDPO (Lai et al., 2024), we introduce a custom difficulty scoring method for each dataset, based on metadata or annotation attributes. The resulting score allows us to categorize samples into *easy*, *medium*, and *hard* subsets in a more interpretable way .

EPFL preference pairs : We define a difficulty score based on the four annotated attributes (correctness, clarity, relevance, completeness). For each sample, we count how many of these aspects favor the chosen answer over the rejected one, and class the difficulty of the sample accordingly.

Ultrafeedback : We begin by filtering samples using prompts to retain only STEM-relevant content (see Appendix 5). A list of keyword patterns (e.g. "mathematic") and exact matches (e.g. "AI", "STEM") is used to reduce topics out-of-domain. As it already includes a numerical score for each pair, we compute a difficulty score by taking the difference between the chosen and rejected scores. A smaller difference indicates a harder comparison.

Stackexchange : We use metadata tags to select only STEM-related questions (see Appendix 9). To ensure high-quality responses, we use the score established by the number of upvotes of the community (see Appendix 8). Only the top two answers are considered, and the sample is kept only if both scores are at least $s \geq 4$. The difficulty score is calculated as the difference between them. Smaller differences indicate harder comparisons. Finally, we sample on the remaining data, in a balanced manner across subjects, ensuring underrepresented subject are included. Summary of the dataset and scores are displayed in Appendix 11 and 12.

3.1.2 MCQA data

We introduce **STEMKnowledge** a subset of WikipediaStem. It provides sections of Wikipedia STEM articles. The prompt is defined as the first 15 words of the section of the article, and the completion is the rest of the section. We believe that it would realign the model to STEM content and thus enhance our performance.

We also present **STEMQA**, a combination of several datasets mainly: Aqua-Rat (Ling et al., 2017), ARC (Clark et al., 2018), MATH (Hendrycks et al., 2021), MedMCQA (Pal et al., 2022), MMLU (Hendrycks et al., 2020) and SciQ (Johannes Welbl, 2017). It also consists of Ultrafeedback (Cui et al., 2023) best-annotated review for the questions related to Computer Science subjects. Certain keywords are used to extract the relevant questions, see Appendix Table. 6. We also added questions and choices generated from various StackExchange discussions, see Appendix Table. 10. The questions and choices were generated with GPT-4o-mini, using a key provided by the course. Overall, STEMQA has two possible combinations : the one with rationales and the one with MCQA questions. MMLU questions are more present (70% of the train set), it contains general STEM content. The other datasets bring specific knowledge to the model, see Appendix Table.7. The aim of the dataset is to teach the model how to answer questions correctly about STEM contents.

3.1.3 RAG dataset

To build our retrieval corpus, we identified the following core categories: Mathematics, Computer Science, Chemistry, Biology, Physics, and Engineering.

For Computer Science, we scraped data from

Stack Overflow using StackAPI, filtering by the most upvoted posts under tags corresponding to widely used programming languages. In Engineering, the number of subfields made it impossible to cover every subject. To maximize topical diversity, we incorporated an appended version of Wikipedia STEM articles. For Chemistry, Biology, and Physics, we relied on the SCIQ dataset, and for Math on MATH dataset which has a variety of undergraduate-level problems on various topics like algebra, geometry, and probability. In opposition to STEMQA, we extract the explanations and integrate them with the question if possible to provide context.

3.2 Evaluation method

The evaluation metric used for DPO is the DPO accuracy (see Appendix A.5, which measures how often the model prefers the chosen response over the rejected one, relative to a reference model. This is defined as the proportion of samples where the chosen reward is greater than the rejected reward. The reward is computed using the log-probability differences between the fine-tuned policy and the fixed reference model.

We evaluate our models based on MCQA **accuracy** with the light-eval method. To evaluate various training alternatives, we evaluate on the test split of STEMQA with single and multi-token prediction. The final model is evaluated on AquaRat, SAT-Math, and LogiQA-en, which are used in the AGI Eval benchmark (Zhong et al., 2023). Those datasets are about Math and Logic, for this reason, we also evaluate on STEMQA which covers a broader palette of subjects. We believe that it reflects if the model learned to answer STEM subjects MCQAs.

3.3 Baselines

Our baseline consists of the Qwen3-0.6B-Base and Qwen2-7B-Instruct (qwe, 2024). The latter having more parameters, it is expected to have better results; therefore, it is used to show the scores an LLM could obtain on those datasets.

3.4 Experimental details

For memory saving purposes we used Flash Attention for an optimized attention computation for MCQA models. We also enabled gradient checkpointing to save memory. The models were run with a batch size of 1, and gradient check-pointing set to 8 or 16 to emulate larger batch sizes.

A first SFT (SFT1) was applied using STEM-Knowledge. We used a learning rate of 1e-6, a linear scheduler for a duration of 3 epochs, and a context size of 4096 tokens. The aim is to align the pre-trained model with STEM knowledge.

3.4.1 DPO experiments

For the reward model, we experiment with different training configurations. With the exception of the SFT1-based experiment, all others starts from the Qwen3-0.6B-Base. We compare two training approaches: the **classic DPO**, trained on the entire dataset in a single pass, and the **sDPO**, trained in a curriculum learning approach over sequential data subsets of increasing complexity. We also evaluate several sDPO variants with modified training hyparameters.

To facilitate comparison, we introduce a baseline model sDPO_ref, trained using a set of reference hyperparameters, detailed in Appendix A.4. We then explore modifications to hyperparameters, producing new variants named according to the changed parameter(s). We aim to find the most effective configuration to align with human preferences. For the final model selection, we compare the sDPO variants with the classic DPO, and the sDPO fined-tuned starting from the SFT1 model.

3.4.2 MCQA model experiments

For this stage, the main experiment consisted in the SFT of the following prompt: `Template + [question] + i) [options] + "Answer:"` and a completion consisting in `i) [correct answer]`. This is referred to as SFT3-A. We also test without the [correct answer] part of the answer for better single token prediction.

Furthermore, we tested a SFT consisting of the question with the same template and as completion the rationale, further referred as SFT3-B. Due to training times, it only uses 25% of the data.

Finally, we created four sequences of tokens consisting of the prompt of SFT3-A with each time one of the four options as the completion. The label is the correct answer. The model computes the probability of each option before applying the CrossEntropyLoss. This is referred to as SFT3-C.

SFT3-A and SFT3-C were implemented using STEMQA, with the rows which contain choices. In opposition, SFT3-B uses only the questions which had a rationale. Furthermore, SFT3-C uses a context length of 2048 (versus 4096 for the other models) due to OOM errors. The MCQA model training

was done on the base model, to get the best training before implemented on the DPO model.

3.4.3 PTQ experiments

Starting from the BF16 checkpoint, we use GPTQModifier to quantize all `Linear` layers except `lm_head`, with symmetric per-channel weight quantization and per-tensor activation clipping. We use a `W8A8` scheme and calibrate on 1,024 examples with a maximum sequence length of 2,048. To find the best memory–accuracy trade-off, we benchmarked seven quantization settings: W8A8, W8A8-1024, W8A8-2048, W4A16, W4A16-1024, W4A4, and W4A4-1024. Each quantized model is evaluated on a subset of 100 MCQA prompts to measure both accuracy and average peak VRAM usage.

3.4.4 RAG experiments

We begin by evaluating combinations of similarity functions (cosine, dot product, max inner product, jaccard) and the number of retrieved documents using the all-MiniLM-L6-v2 model. This model is chosen for its good performance-to-size ratio, as highlighted in the benchmark. We initially tried using Qwen3-Embedding-8B and Linq-Embed-Mistral, respectively 1st and 4th for retrieval and STS on the MTEB leaderboard but they were too slow to be practical. We set the batch size to 100,000 to ensure all documents are embedded and indexed.

We then test this optimal setup with additional models: all-mpnet-base-v2, a larger model known for superior performance (see benchmark), and paraphrase-MiniLM-L6-v2, a smaller alternative with lower performance but faster inference.

We attempted to fine-tune the embedding model by pairing context passages from the retrieval corpus with MCQA training questions, using semantic search to generate positive and hard negative examples. We experimented with both MultipleNegativesRankingLoss and TripletLoss, using Qwen3-0.6B-Base as the main model. However, the final performance was slightly lower compared to our model. As a result, the fine-tuned model was not included in the final pipeline.

3.5 Results

The results about the different stages can be found on Table 1. The different models are capable of increasing the performance of the base model on our benchmark. However, this increase in performance remains in the standard deviation range of the base model, thus making the improvement not statistically significant. We believe that it shows the importance of the base model. Finally, the instructed model shows that LLMs are capable of getting better results, and thus that there is room for improvement[1].

First, SFT1 obtains similar results as the base model. We believe that it would be useful to train the final DPO model, as it has seen STEM content again. The final pipeline, therefore, includes SFT1, DPO and SFT3.

Table 2 presents the performance accuracy (%) of various sDPO variants across difficulty levels. Among all models, sDPO_lr and sDPO_ep_lr_beta achieve the highest accuracies across all difficulty levels. In contrast, sDPO_SFT1, starting from the SFT1 checkpoint shows the lowest performance across all difficulty levels. Similarly, sDPO_beta with a lower β value has a drop of performance compared to the reference. The classic model DPO_classic is behind most sDPO variants, especially on harder examples, highlighting strengths of stepwiseDPO to learn in a curriculum approach. Thus the final chosen reward model is **sDPO_lr**. Results of different training stage are detailed in Appendix 1, 2, and 3, highlighting that models learn much less in hard stage.

Furthermore, SFT3-A, SFT3-B and SFT3-C obtained a log-likelihood accuracy of 50.59% , 24.27% and 51.59% for a single token prediction. For multi-token prediction they obtained a score of 50.47%, 45.53% and 51.47% respectively[2]. (The standard deviation being ±1.21%). Based on this results we decided to report only the single token accuracy. Therefore, SFT3-C, is the final SFT3 model. It performs better than the preceding models displaying that the model learned to answer MCQAs on the datasets.

Among the quantization schemes evaluated (Table 3), the W8A8-1024 configuration (8-bit weights and activations with 1024 calibration samples) provided the optimal balance between accuracy and memory efficiency. This configuration slightly outperformed the full-precision model (42.0% vs. 41.0%) and achieved the highest accuracy-to-VRAM ratio (0.36), indicating the

[1] Instruct has 10 times the number of parameters, it is expected to obtain better results.

[2] Note that SFT3-B and SFT_balanced_choices were trained without the space in front of the label of the choice ("X" and not " X"), to reproduce the results this has to be taken into account when evaluating.

Model	STEMQA	Aqua-Rat	SAT-Math	LogiQA-en
Qwen3-0.6B Base	50.24% ± 1.21%	28.35%±2.83%	35.91%± 3.24%	30.26%± 1.80%
Qwen2-7B-Instruct	65.04%±1.16%	35.43%±3.01%	54.09%±3.37%	37.17%±1.90%
SFT1	50.00% ± 1.21%	27.56%±2.81%	36.36%±3.25%	30.88%±1.81%
DPO	49.24%± 1.21%	23.23%±2.65%	36.82%± 3.26%	29.95%±1.80%
SFT3 (SFT3-C)	51.59%±1.21%	29.92%±2.88%	40%±3.31%	30.41%±1.80%
DPO + SFT3	**51.59%±1.21%**	29.92%±2.88%	38.64%± 3.29%	**31.03%±1.81%**
DPO + SFT3 + W8A8 (1024)	51.12%±1.21%	25.59%±2.74%	36.68%± 3.25%	23.35%±1.66%
DPO + SFT3 + RAG	50.53%±1.21%	**31.15%±2.92%**	**40.91**%±3.32%	30.88%±1.81%

Table 1: Accuracy results with the various variants of fine-tuned model. We evaluate on the single token prediction, as it was a better score in most experiments. Metric is the log-likelihood accuracy. FinesHerbesMCQA corresponds to DPO + SFT3. The bold values represent the best scoring model that we fine-tuned.

Model	**Easy**	**Medium**	**Hard**
sDPO_ref	64.2	62.6	58.0
sDPO_SFT1	57.6	56.7	52.1
DPO_classic	58.6	55.2	50.0
sDPO_lr	70.7	66.2	61.2
sDPO_grad	64.3	62.0	57.3
sDPO_beta	61.5	61.8	57.5
sDPO_ep$_l r$	60.0	59.6	53.6
sDPO_ep$_l r_b eta$	70.1	66.5	62.2

Table 2: Accuracy (%) of fully trained sDPO variants compared to the Qwen3-0.6B-Base reference across validation sets of increasing difficulty. For model parameters and barplot see Appendix A.4 and 4.

Quantization	Accuracy	Avg Peak VRAM (GiB)	Acc / VRAM
W8A8-1024	0.42 ± 0.0496	1.17 ± 0.01	0.36
MCQA model	0.41 ± 0.0494	1.16 ± 0.00	0.35
W4A16	0.41 ± 0.0494	1.37 ± 0.01	0.30
W8A8-2048	0.42 ± 0.0496	2.00 ± 0.01	0.21
W4A16-1024	0.41 ± 0.0494	2.41 ± 0.01	0.17
W8A8	0.41 ± 0.0494	4.89 ± 0.01	0.08
W4A4	0.23 ± 0.0423	3.24 ± 0.01	0.07
W4A4-1024	0.24 ± 0.0429	4.06 ± 0.01	0.06

Table 3: Accuracy vs. VRAM usage over 100 STEMQA questions

most favorable trade-off. Aggressive quantization schemes such as W4A4 exhibited significant accuracy losses (~18–19 pp), while configurations like W4A16 and W8A8-2048 increased memory usage without performance benefits. We thus selected W8A8-1024 for further evaluation on the AGI Eval benchmark.

Finally, we tested the RAG. The base model, using an earlier iteration of the MCQA evaluation dataset, had a base accuracy of 48.24%± 1.21%.

As shown in table 4, even though it is very close with cosine similarity, the confidence intervals overlap with those of the max inner product, meaning the improvement is not statistically significant. However, the max inner product with k = 7 has the highest mean accuracy at 50.17 % ± 1.21, making it the best candidate under our current experimental setup. To assess whether performance differences are statistically significant, we would recommend

k \ s.f.	**cos.**	**max.**	**dot.**	**jac.**
3		49.41 ± 1.21		
4	49.71 ± 1.21			49.59 ± 1.21
5	49.82 ± 1.21	49.65 ± 1.21	48.64 ± 1.21	47.94 ± 1.21
6	49.94 ± 1.21			48.59 ± 1.21
7	49.18 ± 1.21	50.17 ± 1.21	48.79 ± 1.21	48.88 ± 1.21
8		49.41 ± 1.21		49.59 ± 1.21
10				48.88 ± 1.21

Table 4: Accuracy (%) depending on the similarity function and number of documents retrieved

doing a paired t-test on per-question predictions in the future.

We then used all-mpnet-base-v2 and , getting respectively a score of 49.82% ± 1.21% and 49.06% ± 1.21%. This suggests that optimal hyperparameters such as k and similarity function may depend significantly on the embedding space.

4 Analysis

For DPO, Table 2 confirms that the data partitioning is meaningful. Models performs worse and learn less on the hard subset than the medium and easy one, indicating increased difficulty. However, it raises the question whether the hard subset may be too challenging for effective learning. The results also show that sDPO outperforms standard DPO. The curriculum learning approach is thus beneficial for the model to improve alignment. Surprisingly, training from the SFT1 checkpoint degrades the results, possibly due to the its limited training context size.

SFT3-B showed that having a rationale may not enhance the results. The different completion (due to the rationale) may have confused the model, while a single completion (SFT3-A, SFT3-C) displayed better results.

Furthermore, we evaluated single against multi-token prediction. When evaluating on single-token prediction, we obtained respectively 51.18% and 50.82%. For the multi-token prediction accuracy,

we obtained 47.00% and 50.89% respectively (standard deviation of ±1.21%). Training was performed on 20% of data with 1 epoch. With these results, we note that single-token is slightly better; however, both are in the standard deviation range. As the multi-token drastically outperforms the single-token model in multi-token prediction we decided to continue with multi-token prediction.

By looking at the generations, set at default values, see Appendix. A.3, we noticed that even though the completion was just the option, the model would continue the text. It would either create a new question, or talk more about the subject. The generation may stop much later and can get repetitive. We believe that this comes from the SFT training as it works similarly to perplexity, which favors repetitive patterns.

Furthermore, we balanced the labels of the train set (as many labels from each option). We obtained 24.79% accuracy on our dataset[2]. The model preferred to learn that each option has an equal chance to be correct, thus we preferred to stay with a dataset which had approximately 23-27% ratio per label.

Finally, the W8A8-1024 quantized model demonstrated strong performance across multiple benchmarks, including STEMQA (51.12%), Aqua-Rat (25.59%), SAT-Math (36.68%), and LogiQA-en (23.35%), maintaining a high accuracy-to-VRAM ratio of 0.36. Its robustness on complex reasoning tasks like STEMQA and SAT-Math stood in contrast to lower-bit configurations (e.g., W4A4), which frequently produced contextually inaccurate responses. Consequently, the W8A8-1024 configuration offers a suitable memory-efficient solution for deployment on edge devices or student-grade hardware.

5 Ethical considerations

LLMs and RAG models present transformative educational opportunities, especially in STEM education, but also raise social, environmental, and ethical challenges that require proactive mitigation strategies (Bender et al., 2021; Askell et al., 2021).

Our model is primarily English-based due to the pre-training distribution bias in foundation models. Adapting to high-resource languages like French or German necessitates modifications in tokenizer coverage, embedding layers, and multilingual fine-tuning using techniques such as DPO or instruction tuning (Ouyang, 2022; Liang, 2022). For low-resource languages like Urdu or Swahili, these challenges are magnified and require synthetic data, cross-lingual supervision, and rigorous fairness audits (Ramesh et al., 2023; Deya et al., 2024).

Quantization allows model deployment on edge devices and supports digital equity by reducing the inference cost (Yang et al., 2024). However, compression may introduce disparities in output quality across dialects or knowledge domains, exacerbating existing representation gaps (Gao et al., 2024; Hong et al., 2024). Fairness-aware compression and targeted validation are thus essential.

Stakeholders benefiting from the system include students and educators with limited compute access, but risks include potential misuse (e.g., cheating assistance, misinformation), privacy violations, and adversarial manipulation of outputs (Peng et al., 2024). These risks disproportionately affect marginalized groups, necessitating the inclusion of domain-specific resources and community feedback channels (Prabhumoye and Black, 2020).

Mitigation strategies include red-teaming, watermarking, fairness-sensitive quantization, and explainability modules (Guan et al., 2024; Huang et al., 2024). Sustainability is also prioritized through sparse quantization, emission tracking, and model-sharing protocols (Yang et al., 2024).

Ultimately, responsible LLM deployment hinges on transparency, robust evaluation, and value alignment with diverse user needs, especially in educational contexts where trust and accessibility are paramount.

6 Conclusion

In this project, we explored several methods to improve model performance. Our in-house models are able to outperform the base one, showing signs of effective learning during training.

Our approach also revealed some limitations as most gains fell within the base model's variance, limiting their impact. The DPO improved alignment but underperformed compared to combined strategies. Quantization saved memory but hurt reasoning tasks like SAT-Math. The RAG results were inconsistent and sensitive to settings. SFT3-C was constrained by a shorter context window, and STEMQA's MMLU bias may limit generalization.

For future work, integrating the best DPO model in the pipeline could enhance the results.

References

2024. Qwen2 technical report.

Amanda Askell, Yuntao Bai, Anna Chen, et al. 2021. A general language assistant as a laboratory for alignment. *arXiv preprint*.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? *FAccT*.

Beatriz Borges, Negar Foroutan, Deniz Bayazit, Anna Sotnikova, Syrielle Montariol, Tanya Nazaretsky, Mohammadreza Banaei, Alireza Sakhaeirad, Philippe Servant, Seyed Parsa Neshaei, et al. 2024. Could chatgpt get an engineering degree? evaluating higher education vulnerability to ai assistants. *Proceedings of the National Academy of Sciences*, 121(49):e2414955121.

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*.

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

Krishno Deya, Prerona Tarannum, Md. Arid Hasan, Imran Razzak, and Usman Naseem. 2024. Better to ask in english: Evaluation of large language models on english, low-resource and cross-lingual settings. *arXiv preprint*.

Adam Fisch, Jacob Eisenstein, Vicky Zayats, Alekh Agarwal, Ahmad Beirami, Chirag Nagpal, Pete Shaw, and Jonathan Berant. 2025. Robust preference optimization through reward model distillation.

Mingqi Gao, Xinyu Hu, Xunjian Yin, Jie Ruan, Xiao Pu, and Xiaojun Wan. 2025. Llm-based nlg evaluation: Current status and challenges. *Computational Linguistics*, pages 1–28.

Yunfan Gao, Jiacheng Li, Weizhu Zhang, et al. 2024. Towards fairness-aware model compression. *arXiv preprint*.

Melody Y. Guan, Manas Joglekar, Eric Wallace, et al. 2024. Deliberative alignment: Reasoning enables safer language models. *arXiv preprint*.

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

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*.

Junyuan Hong, Jinhao Duan, Chenhui Zhang, et al. 2024. Decoding compressed trust: Scrutinizing the trustworthiness of efficient llms under compression. *arXiv preprint*.

Yue Huang, Lichao Sun, Haoran Wang, et al. 2024. Trustllm: Trustworthiness in large language models. *arXiv preprint*.

Matt Gardner Johannes Welbl, Nelson F. Liu. 2017. Crowdsourcing multiple choice science questions.

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. 2024. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms.

Percy et al. Liang. 2022. Holistic evaluation of language models. *arXiv preprint*.

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*.

Long et al. Ouyang. 2022. Training language models to follow instructions with human feedback. *arXiv preprint*.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. 2022. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Proceedings of the Conference on Health, Inference, and Learning*, volume 174 of *Proceedings of Machine Learning Research*, pages 248–260. PMLR.

Benji Peng, Keyu Chen, et al. 2024. Jailbreaking and mitigation of vulnerabilities in large language models. *arXiv preprint*.

Shrimai Prabhumoye and Alan W Black. 2020. Case study: Deontological ethics in nlp. *arXiv preprint*.

Krithika Ramesh, Sunayana Sitaram, and Monojit Choudhury. 2023. Fairness in language models beyond english: Gaps and challenges. *Findings of ACL*.

Qwen Team. 2025. Qwen3 technical report.

Amir Pouran Ben Veyseh, Viet Lai, Franck Dernoncourt, and Thien Huu Nguyen. 2021. Unleash gpt-2 power for event detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6271–6282.

Kevin Yang, Deepak Narayanan, Yian Zhang, et al. 2024. Llm compression: A benchmark for efficient language model inference. *arXiv preprint*.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. 2023. Agieval: A human-centric benchmark for evaluating foundation models.

A Appendix

A.1 Contributions statement

All members contributed roughly equally to the project, the different milestones, and the final report writing. Gustave Lapierre was responsible for the RAG model, Lucie Huamani-Cantrelle developed the quantized model, Jehan Piaget implemented the DPO model, and Valentin Perret worked on the MCQA model and the SFT1 training.

A.2 AI policy coverage

For the reward model, ChatGPT helped write the files for plotting the metrics and part of the main code, mainly to log the output and store the metrics during training. It assisted in optimizing parameters and commenting the code. It also helped fill the STEM subject and keywords when filtering StackExchange and UltraFeedback data, which were manually checked and completed.

For the MCQA model, ChatGPT produced many parts of code: for example, for the SFT trainer, it helped to set it up properly. If we wanted to add something, we would often ask it on Google and ChatGPT, and try the solution. To verify if the implementation was correct we also looked at the documentation to double-check the information. We also resorted to the good old *print(value)* to verify the implementations. Looking at the loss plots proved very insightful, they showcased if what was implemented worked or not. We also used it to obtain the relevant information from the different datasets and format it into one single dataset.

ChatGPT-4o was used to assist with debugging and optimizing memory profiling during the development of the quantized model. It helped identify and resolve configuration issues in the quantization pipeline, and streamlined scripts for evaluating VRAM usage.

Regarding the RAG, ChatGPT 4o was used to create most of the basic functions in modify_datasets and to help optimize training and execution time.

A.3 Example of text generation of the model

```
"Answer: D. momentum is larger.Human beings are born with [...] True or false? Answer: True.Human beings are born w [...]"
```

A.4 Reward model hyperparameters details

The sDPO_ref model is trained with the following hyperparameters:

- Starting model: Qwen3-0.6B-Base
- Epoch: 1
- Gradient accumulation steps: 8
- Train batch size: 1,
- Learning rate: 5e-6
- Beta: 0.1
- Max length (prompt + completion): 1024
- Max prompt length: 512
- Use flash: False
- Use gradient checkpointing: True
- Learning scheduler type: cosine
- Warmup ratio: 0.1
- Weight decay: 0.01
- Gradient clipping norm: 1

All variants are based on the same configuration, with specific changes as follows:

- DPO_classic Trained on the full dataset in a single pass
- sDPO_SFT1 Starting model: SFT1
- sDPO_lr Learning rate: 2e-5
- sDPO_beta Beta: 0.05
- sDPO_grad Gradient clipping norm: 10
- sDPO_ep_lr Epochs: 2, learning rate: 2e-6
- sDPO_ep_lr_beta Epochs: 3, learning rate: 2e-6, beta: 0.05

A.5 Reward model accuracy metric

Accuracy is defined as the proportion of sample where chosen_reward > rejected_reward, with :

$$chosen\_reward = \log p_{pol}(c) - \log p_{ref}(c)$$

$$rejected\_reward = \log p_{pol}(r) - \log p_{ref}(r)$$

With pol: fine-tuned policy, ref: frozen, reference policy, c: chosen answer, and r: rejected answer. This metric reflects whether the fine-tuned policy learned successfully to favor the preferred answer.

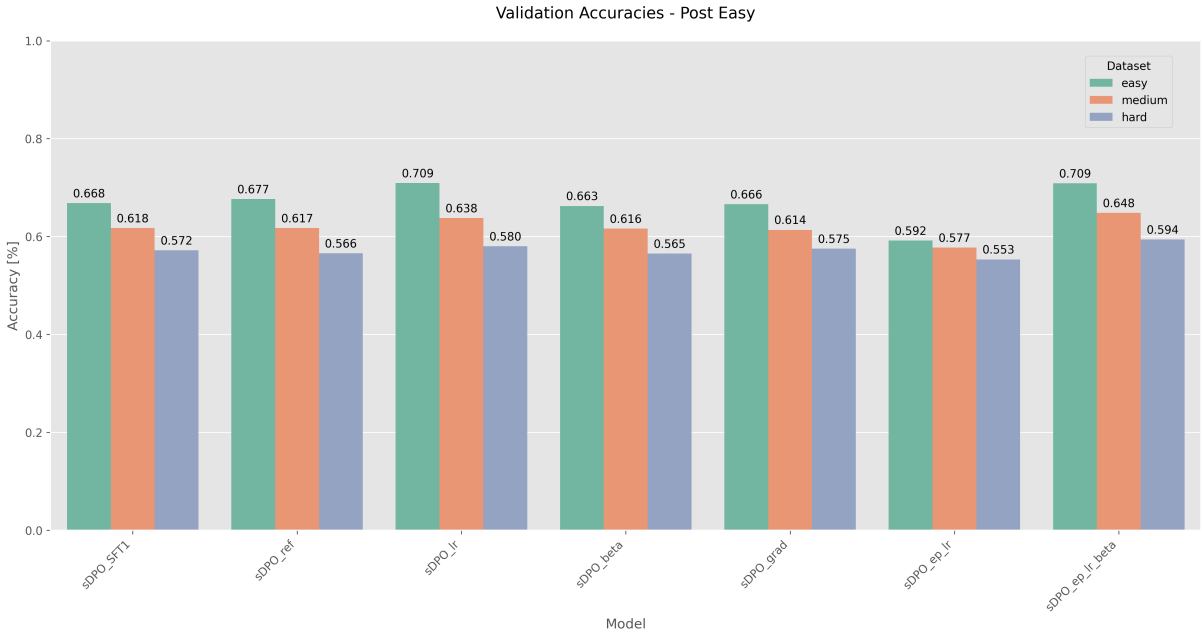

Figure 1: Post-easy training stage accuracy (%) of the different DPO models compared to Qwen3-0.6B-Base on easy, medium, and hard validation sets.

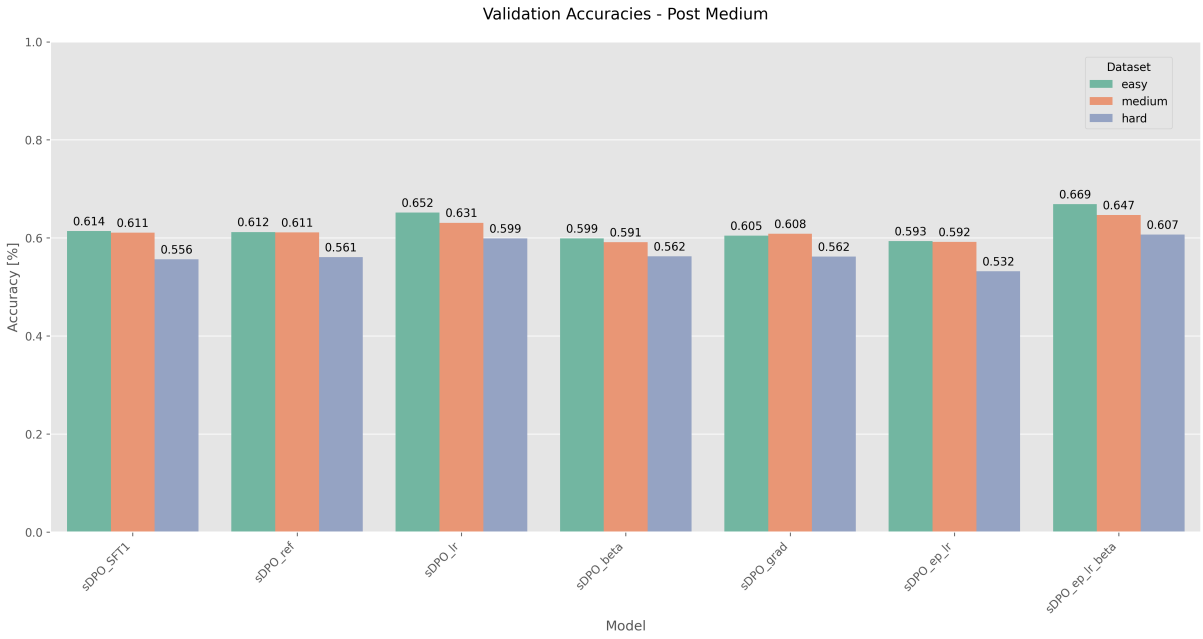

Figure 2: Post-medium training stage accuracy (%) of the different DPO models compared to their previous model (post-easy training model) on easy, medium, and hard validation sets.

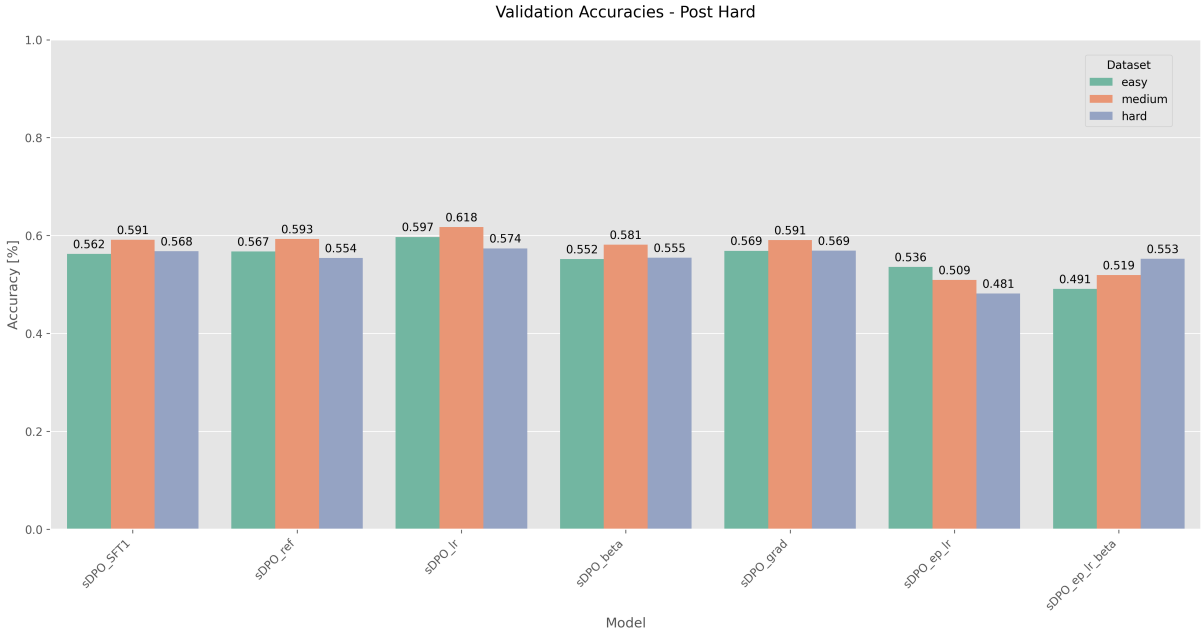

Figure 3: Post-hard training stage accuracy (%) of the different DPO models compared to their previous model (post-medium training model) on easy, medium, and hard validation sets.

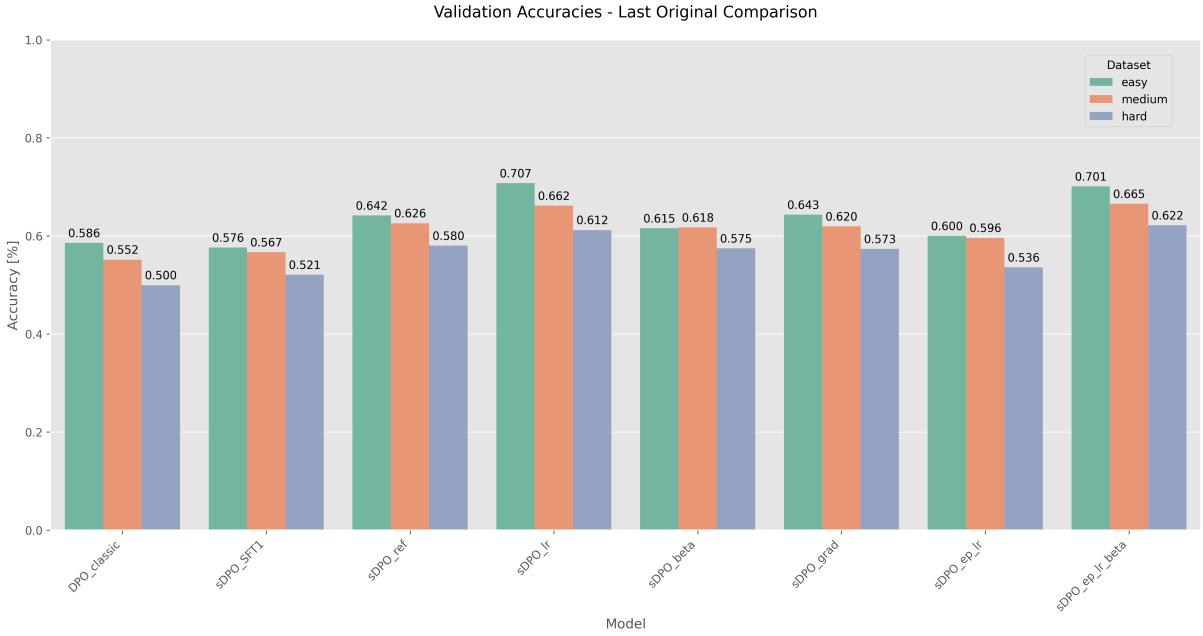

Figure 4: Post-hard training stage accuracy (%) of the different DPO models compared to Qwen3-0.6B-Base on easy, medium, and hard validation sets. Reflects performance of the fully trained models compared to the original ones.

List of STEM Keywords for filtering UltraFeedback dataset

Exact Match Keywords

STEM, AI, int, RNA

Pattern Matching (Regular Expression)

science, technology, engineering, math, scientific, technical, algebra, calculus, geometry, trigonometry, statistics, probability, derivative, integral, equation, matrix, theorem, vector, logarithm, formula, physics, quantum, mecha, relativity, thermodynamics, optics, force, velocity, acceleration, mass, energy, momentum, gravity, electricity, magnetism, circuit, resistor, capacitor, neuron, machine learning, data analysis, data visualization, data science, acid-base, algorithm, computer, programming, python, java, C++, neural, network, chemistry, atom, molecule, reaction, compound, organic, inorganic, chemical, javascript, html, equilibrium, stoichiometry, biolog, genetics, DNA, evolution, organism, protein, enzyme, photosynthesis, microscope, mitosis, bacteria, fluid, SQL, database, code, boolean, float, string, print, step by step, astronomy, medicine, aerospace, psychology, environment, electric, magnetic, thermo, cyber, brain, complexity, artificial, intelligence, model, token, GPU, NLP, LLM, civil, robot, material, biomedical, biochemistry, biophysics, biotechnology, bioinformatic, simulation, computation, numerical, deep learning, reinforcement learning, proof, axiom, lemma, corollary, nuclear, astrophysics, cosmology, photon, electron, proton, neutron, software, hardware, compiler, syntax, debug, loop, recursion, hash, encryption, compression, tensor, pytorch, keras, regression, classification, blockchain, crypto, informatic, aerodynamics, hydraulics, structural, entropy, kinematics, dynamics, friction, torque, polynomial, exponential, factorial

Table 5: Keywords and patterns used to filter STEM-relevant samples from the UltraFeedback dataset. Exact match keywords uses word boundaries for exact term identification to avoid false positive, while pattern matching are searched as substrings within the prompt text.

List of STEM Keywords for filtering UltraFeedback dataset for the STEMQA dataset

Exact Match Keywords

STEM, AI

Pattern Matching (Regular Expression)

SQL, database, Python, Java, C++, C, JavaScript, pandas, NumPy, API, recursion, complexity, OOP, debug, syntax, Rust, Go, R, MATLAB, TensorFlow, PyTorch

Table 6: Keywords and patterns used to filter STEM-relevant samples from the UltraFeedback dataset. Exact match keywords uses word boundaries for exact term identification to avoid false positive, while pattern matching are searched as substrings within the prompt text.

Dataset	Rationale	MCQA	Subject	Train Category	Used at testing	Lines in training
Aqua-Rat	Yes	Yes	Algebra	SFT3-A/B/C	Yes	12000
ARC	No	Yes	Science	SFT3-A/C	Yes	1118
MATH	Yes	No	Math	SFT3-B	No	2000
MEDMCQA	Yes	Yes	Medical	SFT3-A/B/C	Yes	12000
MMLU	No	Yes	STEM	SFT3-A/C	Yes	84000
Stack Overflow	Yes	Yes	STEM	SFT3-A/B/C	Yes	819
SciQ	Yes	Yes	Science	SFT3-A/B/C	Yes	6000
UltraFeedback	Yes	No	Computer Science	SFT3-B	No	1980

Table 7: Breakdown of the various datasets in STEMQA. It tells us if a dataset contains a rationale, and if it is in an MCQA format. Furthermore, we provide the training for which it was used.

Score	Min. Upvotes (Not Accepted)	Min. Upvotes (Accepted)
-1	< 0	< 0
0	0	0
1	1	0
2	3	1
3	7	3
4	15	7
5	31	15
6	63	31
7	127	63
8	255	127
9	511	255
10	1023	511

Table 8: Minimal number of upvotes required to reach a certain score in the StackExchange-preferences dataset. A distinction is made between answers that were accepted by the questioner and those that were not.

List of STEM topics retained for the StackExchange dataset

`stackoverflow, math, physics, chemistry, biology, cs, stats, datascience, robotics, electronics, engineering, quantumcomputing, scicomp, bioinformatics, ai, cstheory, mathoverflow.net, codereview, softwareengineering, dsp, space, astronomy, earthscience, quant, mathematica, raspberrypi, arduino, networkengineering, reverseengineering, security, 3dprinting, cogsci, computergraphics, devops, drones, gis, iot, materials, mechanics, retrocomputing, sound, sqa, cseducators, matheducators`

Table 9: List of STEM-related topics selected to filter the StackExchange datasset.

- Math
- Computer Science
- Statistics
- Data Science
- Artificial Inteligence
- Bioinformatics
- Computationnal Science
- Math Overflow
- Computer Science Theory

Table 10: List of the Stack Exchange forums used to generate questions coming from the Stack Overflow forum.

Dataset	Original	After STEM-filtering	After remaining filtering	Final
StackExchange	10,807,695	1,040,046	36,443	25,000
UltraFeedback	60,917	22,738	22,734	22,734
EPFL Preferences	24,365	24,365	24,240	24,240
Combined	–	–	–	71,974

Table 11: DPO dataset filtering and sampling process. The table shows: (1) original sample counts, (2) samples remaining after STEM subject filtering (StackExchange and UltraFeedback only), (3) samples after quality filtering (removing preference errors, empty answers, and low-quality samples), and (4) final sample counts after downsampling StackExchange to 25,000 to balance between dataset sources. The combined dataset of 71,974 samples is then split equally into easy, medium, and hard difficulty subsets. Each subset uses a 90/10 train/validation split while preserving the proportion of samples from each source dataset (stratified split).

Dataset	Possible Range	Hard	Medium	Easy
StackExchange	1 – 6	1 – 1	1 – 2	2 – 6
UltraFeedback	0.25 – 4	0.25 – 0.75	0.75 – 1.5	0.75 – 1.5
EPFL	-4 – 4	-4 – 1	1 – 2	2 – 4

Table 12: Score distribution ranges across difficulty partitions for each dataset. Easy partition contains samples with clear preference (high score differences), Medium partition contains samples with moderate preference, and Hard partition contains samples with ambiguous preference (low score differences).