How to replace TAs : A comprehensive study on letting LLMs answer your questions

Anthony Kalaydjian | 370837 | anthony.kalaydjian@epfl.ch
Anton Balykov | 368883 | anton.balykov@epfl.ch
Eric Saikali | 326450 | eric.saikali@epfl.ch
ShAIkerspear

Abstract

This project aims to address the challenge of creating an AI model capable of answering exam questions by focusing on answering MCQA exams questions. Findings highlight that while the model can handle general Q&A realatively easily, it struggles with more specific questions, especially when they deal with mathematics. Several models were trained using Supervised Fine Tuning (SFT) as well as Direct Preference Optimization (DPO) and evaluated on several test datasets. The results show a way better alignment of the model with preference data after training it using DPO while improving slightly its performance. The best model was also quantized to 8 & 4 bits precisions whose performances have been compared. General results show that answering STEM MCQAs is very challenging, with results that are not ideal.

1 Introduction

Large Language Models (LLMs) have shown the beginning of a revolution in many sectors, with the advent of closed source models such as ChatGPT which was just integrated in the Apple's Apple Intelligence system, Claude as well as open source alternatives such as Mistral AI's models or Meta's LLAMA. As time passes, the performance of these models and their range of skills increases. As such, multi-modal models and API-augmented models are starting to surge in our phones with better voice assistants and computers with coding assistants, as well as in other tools.

Nonetheless, one sector that lacks behind when it comes to LLM integration is education. Even though LLMs are getting better and better, they have yet to match proper professor/expert level in some education fields, especially in science, technology, engineering, and math (STEM) related domains. But, as for the other sectors, having educative LLMs could heavily reduce the resources required for teaching and improve it by enabling students to access a 24/7 tutor in the palm of their hands.

In this optic, this paper focuses on creating a language model specialized in answering EPFL MCQA exam questions. Question answering is one of the most used ways of interacting with a LLM system and has been democratized with the appearance of systems such as ChatGPT, Gemini etc. Although these models are able to provide coherent answers most of the time, this task remains challenging. And it is even more so when the questions deal with science as they usually require some sort of reasoning on the question and specific knowledge. This is the case that is tackled here as the main goal is to create a system that answers STEM related MCQAs.

The approach of this project revolves around 3 main parts. The first part tackles data collection, with the generation of a custom DPO training set from synthetically generated answers to EPFL exam question which have been evaluated by EPFL students, as well as carefully selected available SFT and DPO datasets. These additional datasets enable furtherfine-tuning as they have different distributions and provide wider range of information to train on. The second part tackles the fine-tuning of the Phi-2 model according to a well defined strategy. After the training, the last part deals with evaluating the performance of the trained model when compared to a baseline.

As the Phi-2 model is quite big one and takes a lot of space both for storing it and during inference, an approach to reduce its size (named quantization) was chosen as a way to mitigate this issue. This allows the model to become smaller and run faster, allowing its usage on a broader range of devices.

2 Related Work

Different question answering LLMs and systems have been developed throughout the last couple of years. Some of them try to specialize in a par-

ticular domain such as Codellama (Rozière et al., 2024) which focuses on code generation or Meditron (Chen et al., 2023) which specializes in medical question answering while others remain general such as GPT3 (Brown et al., 2020). In this work, the goal is to train a model to answer MCQAs which cover a relatively broad gamut of fields.

Choosing to have a specialized or a more general purpose LLM yields to make a choice between having a model that performs better on a very specific task, but poorly on others or having a model that performs well on any task but has lower performance when compared to the former on its specialization field. In this matter works have shown how to strike a balance between the two (Zhang et al., 2023) and inspire the design choices that will be made here to yield a model capable of answering a wide range of STEM questions. In this matter, the model will be trained on both general STEM MCQAs as well as more domain specific (Maths in this case) ones.

Given a specific task with a dataset, (Yigit and Amasyali, 2023) shows that first training sequentially on different datasets that encounter the same task before training on the target dataset yields better results than simply training on the target dataset.

For instance, Meditron (Chen et al., 2023), which is a model developed for answering medical questions, was trained in a similar manner, on a broad variety of medical MCQA datasets with different scales and granularities. This model shows good results on many medical benchmarks. Therefore, the strategy that has been chosen has already proven to be performant in a larger scale model (7B & 70B).

Pairing therefore the data balancing from (Zhang et al., 2023) with the sequential training from (Yigit and Amasyali, 2023) seems to be an interesting approach whose results will be assessed against other paradigms.

Furthermore; by introducing, for each question, the correct answer as well as a explanation of the answer in the training prompt, the model is expected, at inference time, to use a Chain of Thought reasoning approach (Zhang et al., 2022) to yield its answer which has been proven to improve the model's accuracy.

3 Approach

3.1 Model architecture

Phi-2 (Hughes, 2023) is a transformer based model with next-word prediction objective. It has 2.78B parameters, a context length of 2048 tokens and has been pretrained on 1.4T tokens. The Phi-2 Model architecture consists of several layers, with first an embedding of the tokens going from an a one hot encoding of 51,200 tokens to a vector space of dimension 2,560. This is followed by a succession of 32 decoding layers consisting of self attention modules of same input/output dimension with the addition of rotational embedding which make computations more efficient. This is followed by an MLP with GeLU activations and the following layer sizes : 2560 - 10240 - 2560. The model is completed by a linear head.

The base model has been pretrained on the data on which its predecessors Phi-1.5 and Phi-1 were trained, consisting of subsets of Python codes from The Stack v1.2 (Kocetkov et al., 2022), Q&A content from StackOverflow, competition code from *code_contests* (Li et al., 2022), and synthetic Python textbooks and exercises generated by gpt-3.5-turbo-0301. The training dataset was 250B tokens long consisting of combined NLP synthetic data created by GPT-3.5, and filtered web data from Falcon RefinedWeb (Penedo et al., 2023) and SlimPajama (Shen et al., 2024), both assessed by GPT-4.

3.2 Training architecture

For fast and efficient training purposes, the Transformer Reinforcement Learning library from HuggingFace (von Werra et al., 2020) was used. It allows to define a complex and robust training procedure, utilizing modern training techniques. The base phi-2 model was fine-tuned using LoRA (Hu et al., 2021) with a peft (pef, 2023) adapter, which allow to only train specific layers of the model with low rank approximations matrices to update the weights, which significantly reduces memory consumption and training time.

The LoRA adapter is used with rank 16, $\alpha = 16$ and a dropout of 5% (according to (Hu et al., 2021), these values are reasonable for having much less trainable parameters while still being able to train the model to have reasonably good performance). As a result, the only part of the model that needs to be saved is the adapter itself, which reduces the size of the checkpoint and accelerates transfer times.

3.3 Training Strategy

As stated in section 2, the main training strategy revolves on sequentially training the model on various datasets, going from the most general to the most specific one.

The training consists of two different modes:

Supervise Fine Tuning (SFT) Trains the model to generate proper answers by giving reference answers. The loss used for training the model is the cross-entropy loss which computes the sum of the probabilities of the expected token from the density that is provided by the model and tries to maximize it. SFT is performed in a teacher-forcing manner, at each token, the model is fed all of the previous gold tokens. Its output is then compared to the corresponding gold token using the Cross Entropy loss as follows:

$\mathrm{L}(\mathrm{y}_t, \{y^*_{<t}\}) = -\sum_{t=1}^{T} \log P\left[y_t | \{y^*_{<t}\}\right]$

where

$\mathrm{P}[y_t | \{y^*_{<t}\}] = \frac{e^{S_w/\tau}}{\Sigma_{w' \in V} e^{S'_w/\tau}}$ is the probability density of the prediction of the model given all of the previous tokens.

This loss is computed in parallel thanks to the transformer architecture, and can be minimized using gradient descent and more specifically the adamW optimizer.

Direct Preference Optimization (DPO) Aligns the model to human preference, by training it to prefer the better answer to a given question when provided ranked annotated. As seen in the following DPO objective, the training updates the parameters θ in order to increase the policy the model would give to the better answer while reducing the one of the worse.

$$\mathrm{L}(\pi_\theta; \pi_{ref}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{ref(y_w|x}} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{ref}(y_l|x)} \right) \right]$$

Each dataset is associated to one of the two modes. When training on a given dataset, the associated training is performed.

While most datasets were retrieved from online public sources, one of the datasets that is used for DPO was collected by students from the CS-522 course. Each was assigned 100 MCQAs from EPFL exams to which they had to generate 2 answers using ChatGPT and evaluated both.

Two training phases will be performed. A first one on a pre-processed list of datasets giving insights on a second one which also used a broader list of datasets.

3.4 Quantization

Once trained, the best model will be quantized using the BitsAndBytesConfig from the transformers library in order to reduce its size while keeping most of its performance. The quantization's degree will be assessed by comparing the performance/size ratio with different quantization amounts. The best one will be kept.

4 Experiments

4.1 Data

Fine-tuning the base model was experimented on multiple and diverse datasets, for both Supervised Fine Tuning (SFT) and Direct Preference Optimisation (DPO). These datasets were first processed to reach a consistent format all across.

- The DPO json lines file has the following format ["prompt", "rejected" "choosen"].
- The SFT json lines file has the following format ["id", "subject", "question", "answer" and "answer_text", "choices"].

The *answer_text* and *subject* fields help to store the explanations and subjects of their questions, allowing to improve the answer generation step by adding step by making the model use a step by step reasoning scheme. The *choices* field is also preserved to keep track of the gold answer.

The datasets experimented on for training are the following :

- EPFL dataset : A collection of around 20K/30K preference data pairs, split into two subsets.
  - **EPFL_SFT** : 40% of the *chosen* field of the EPFL data used for SFT training in order to accommodate the model to the distribution of inputs it will infer on.
  - **EPFL_DPO** : used for DPO training, which was processed by keeping 60% of all DPO pairs.
- **MathQA** (SFT) (Aida Amini and Hajishirzi, 2019) : Used to fine tune the model to reason and to answer STEM (science, technology, engineering, and math) questions. Formatting

this dataset is straight forward except for retrieving the *answer_text* as a formatted answer ("Answer : <letter>") is not consistently used. The answer was still retrieved by splitting the text properly and using RegEx.

- **helpSteer** (DPO) (Zhilin Wang, 2023) : Aimed at aligning the model to become more helpful, factually correct and coherent.
- **OpenBookQA** (or OpenQA) (SFT) (Todor Mihaylov, 2018): Used to train the model to answer open-domain questions.
- **ScienceQA** (SFT) (Lu et al., 2022): Used to train the model on more general STEM questions.
- **tal_scq5k** (SFT) (TAL, 2023) : Also used to fine tune the model to reason and to answer STEM questions on a mathematical point of view.
- **all_mcqa** (SFT) : A dataset containing all of the data entries from the previous SFT datasets but this time shuffled.
- **balanced_merged** (SFT) : the all_mcqa but balanced to have the same proportion of answer classes (A,B, C or D).

For testing, the MMLU (Hendrycks et al., 2021) dataset was used as well for further evaluation. It has been processed to have the normalized SFT format.

In addition, the longest data-points (>512 tokens) were removed from all datasets to accelerate training and reduce memory consumption, as most samples (>90%) contain less than 512 tokens. The datasets were also randomly sub-sampled to reduce further their size and accelerate training. Datasets with less than 20k datapoints are kept as is while the ones with more datapoints are clipped to have a 20k datapoints size.

All of these datasets were split into train/test_overfit/test_comparison/test_quantization datasets with a proportion of 50%, 25%, 10%, 15% per dataset respectively.

Test losses are plotted during training using the test_overfit dataset. First fine tuned models are then compared using the test_comparison datasets, while the base and quantized models are finally compared using the test_quantized dataset.

Further considerations of the data are about its usage, while all datasets being used are free of use, the usage of the EPFL dataset presents some ethical repercussion as this data might consist of re-used exam questions which might contain answers. Direct access to this dataset and its usage could be a "dataleak" if the corresponding professor didn't agree for it to be used. Having a chatbot that could answer these questions specifically could also compromise the validity of quizzes.

4.2 Evaluation method

Evaluation is assessed on the MCQA datasets by computing the accuracy of each model on the test set.

For the preference data, accuracy of choosing the preferred alternative is used.

4.3 Baselines

The finetuned and compared models have been compared in parallel with the untrained initial model, Phi-2.

4.4 Experimental details

For all the experiments that, the pretrained Phi-2 model was used as a base model. Trainings is done using the Transformer Reinforcement Learning library from HuggingFace (von Werra et al., 2020). For SFT tasks, the batch size was 4 both for training and evaluation. For the DPO training, the batch size was set to 1 because of the training procedure algorithm and Out-Of-Memory errors that occured with larger batch sizes.

The cosine learning rate scheduler was used for smoother learning rate change and warmup during the trainings. For all datasets except for the EPFL ones, the learning rate of $1e^{-5}$ was used as it was high enough for the model to learn patterns and low enough for it not to diverge. For EPFL data, the learning rate was set to $1e^{-4}$ due to the noise in the collected data.

Trainings took different time depending mainly on the size of datasets, number of epochs per dataset and also the task (DPO took more time due to the fact that it also uses 2 models: policy and reference).

4.4.1 Preliminary training results

A few models were trained based on different training data compositions:

① The base model trained successively on all of the datasets without modification. With a number of trained epochs of: [2, 2, 3, 1, 2]

② The base model trained on the same dataset list without Helpsteer. Number of trained epochs: [2, 2, 4, 3]

③ Same as model 2 but clipping the EPFL datasets to only keeping 2 answers for each DPO question to control the number of appearances of each question. Number of trained epochs: [2, 2, 4, 5]

The base model is the original pretrained Phi-2 model.

For this phase, no additional context information was used. The answer only comprised of the letter answer itself.

Model	EPFL_DPO	EPFL_SFT	HelpSteer	MathQA	OpenQA
①	0,4767	0,5082	0,4596	0,2657	0,7133
②	0,4793	0,5089	0,4686	0,2587	0,72
③	0,4841	0,5023	0,4641	0,2652	0,72
base	0,2225	0,1852	0,3318	0,2624	0,72

Table 1: Preliminary training results' accuracies over 1

The results from table 1 are not great in most test datasets except for OpenQA. Although the trained model barely reaches around 50% accuracy on the EPFL_DPO and EPFL_SFT evaluations, it still shows massive improvements when compare to the base model which got very poor results.

On the other hand, it seems like the model wasn't able to properly learn from the MCQA task. This may be due to two factors. First, the fact that context and answer explanation wasn't yet fed into the model and second, that DPO training might confuse the model and make it forget about its previous SFT training on MCQA datasets.

4.4.2 Further training

Given the insights of the first training phase, it is clear that the crux of the matter lies both on the data's nature as well as on the step at which DPO is performed.

Taking these in consideration, the data has been modified and a few other models were trained with the inclusion of proper explanations in the input prompt. The format used for training is the following:

```
### Question ... ### Explanation ... ### Answer ...
```

The chosen training data configurations are the following:

④ The base model trained successively on all_mcqa and EPFL_DPO. With a number of trained epochs of: [2, 1]

⑤ The base model trained successively on EPFL_DPO, tal_scq5k, MathQA and ScienceQA. With a number of trained epochs of: [2, 3, 2, 1]

⑥ The base model trained on tal_scq5k, MathQA, SicenceQA and EPFL_DPO. Number of trained epochs: [4, 3, 2, 1]

⑦ Same as model 6 but without EPFL_DPO at the end. Number of trained epochs: [4, 3, 2]

⑧ The base model trained successively on balanced_merged and EPFL_DPO. With a number of trained epochs of: [2, 1]

4.5 Results

4.5.1 Fine-tuning results

Once trained, all the models were tested on each of the *test_comparison* datasets. The results of the training are consigned in table 2.

Model	MathQA	OpenQA	ScienceQA	tal	MMMLU	EPFL DPO
④	0,263	0,734	0,227	0,240	0,572	0,586
⑤	0,251	0,756	0,14	0,212	0,613	0,554
⑥	0,262	0,734	0,224	0,25	0,619	0,585
⑦	0,227	0,643	0,110	0,188	0,541	0,546
⑧	0,265	0,734	0,227	0,270	0,617	0,585

Table 2: Training test_comparison results' accuracies (out of 1)

These results first show a clear trend. All of the trained models have a harder time answering math or science related question. This is shown as the accuracy of each wanders at about 0.25, which corresponds to a random model.

Regardless of these, the results also show that model ⑧ is the best performing one, reaching an accuracy of 0.61 on the MMMLU benchmark and having close to the highest accuracies in all of the other test datasets.

This can potentially mean that balancing the data facilitates better results on topic-specific datasets as model does not learn as much bias towards specific answer choices. However, such an approach clearly requires more distilled data.

Plots for the training and testing curves of model ⑧ can be seen in figures 1 and 2

As seen in the training and testing curves, the iteration number was properly set, as the test loss exponentially decays and doesn't go up.

This analysis tends to choose this model as final candidate.

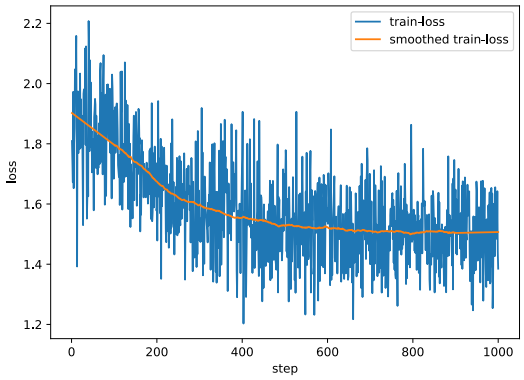

Figure 1: Model ⑧ balanced_merged training loss curve

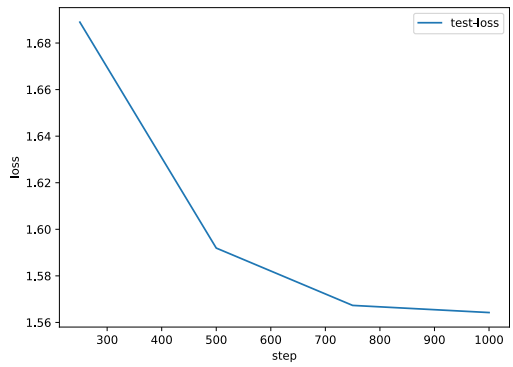

Figure 2: Model ⑧ balanced_merged test curve

5 Analysis

Regarding the first preliminary training, which didn't take into account the explanation of the answers in the input training prompts, it has been seen that the trained models had trouble outputting a single letter in its output. For instance, models had approximatively 150 missing answer cases in the OpenQA dataset.

On a human analysis level, what was interesting during the inference is that sometimes model didn't only answer the given question, but also created its own questions based on the topic of the original one, trying to answer them as well. This behavior could potentially be explained with the fact that the model was pretrained with the following format:

`Instruct: <prompt> Output:`

During finetuning a different format was used to align the model with the desired data. Since the model also has a "chat" mode when it continues the discussion, it probably treated the questions as a chat. The example question from OpenQA dataset figure 4 in the appendix shows how model continued to create follow-up exercises even though this was not required. The answer it gave is correct but then it went deeper into explanations and tried to create more of similar questions with their answers.

Other examples shows that the model goes for a broader explanation arguing and going even further and not limiting itself to the MCQA task format. This is clearly shown by the following example 3 where the model does a valid point while respecting somehow the standards the MCQA by answering none of the above. this can be argued to be demonstration of the "task failed successfully" paradigm.

QUESTION : Members of rock bands often perform with

Options:
A. flutes
B. sandals
C. earplugs
D. gloves

ANSWER :
Solution:
Members of rock bands often perform with instruments such as guitars, drums, and bass. Flutes, sandals, and earplugs are not typically used in rock music performances. Gloves are not commonly used in rock music performances either. Therefore, the correct answer is none of the above.

Figure 3: Example

Regarding MCQA accuracy levels, it is observed that balancing the data helped improving the performance of the model.

6 Ethical considerations

From an ethical point of view, a generator model like the one described above has multiple vulnerabilities imposed upon its stakeholders. These are EPFL students accessing the model, professors and online services. Some threats and attacks are: allowing for repudiation that an exam answer is generated by the model would disable teacher to identify cheating, this renders authenticity obsolete. It could also harm people by compromising the outputs' integrity as they are not sanitized and could have discriminatory biases or incorrect answers. As the model is fine-tuned on course data, malicious people could query help for performing cyber-attacks like denial of service resulting in a breach of availability. It could also breach confidentiality by revealing re-used confidential questions in exams. These attacks are severe as they compromise the legitimacy of a diploma. To mitigate most issues, a second model could be used to check for both harmful inputs and model outputs, and not show the latter when it concerns sensible content. Further vulnerabilities of this model giving potential or

malicious prompts which upon question-answering could tamper with a person's, or entity's preservation like "How to create a bomb?". Other, could be cultural bias induced by the fine-tuning of the model toward one defined truth or even generation of incorrect information and hallucinations.

7 Conclusion

In conclusion, this project demonstrates that it is feasible to develop a 2.8 billion parameter model capable of answering exam MCQA questions, particularly when the questions are general in nature. The model shows potential in learning to answer more STEM oriented questions, highlighting the importance of balanced data for effective training. Remarkably, the model achieved a pretty high score on the EPFL_DPO benchmark, outperforming the initial phi-2 model of 22% accuracy up to 58%, for other benchmarks, more than 2% in various areas such as Math and science where achieved such as 27% for tal_scq5k and 61% on MMLU.

Numerous methodologies employed were learned during this project, including quantization and fine-tuning techniques, used along the utilization of LoRA (Low-Rank Adaptation) to facilitate the model's training, saving time and space meanwhile. High-volume data for both training and testing were conducted despite limited resources, challenges such as handling `torch.cuda.OutOfMemoryError` and optimizing time-wise operations.

However, challenges remain, particularly regarding time limitations for thorough training and the SCITAS cluster's resource constraints for running larger models. Addressing these challenges will be crucial for further advancements and the development of even more capable and robust AI models for educational purposes.

References

2023. State-of-the-art parameter-efficient fine-tuning (peft) methods.

Shanchuan Lin Rik Koncel-Kedziorski Yejin Choi Aida Amini, Saadia Gabriel and Hannaneh Hajishirzi. 2019. Mathqa.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Zeming Chen, Alejandro Hernández Cano, Angelika Romanou, Antoine Bonnet, Kyle Matoba, Francesco Salvi, Matteo Pagliardini, Simin Fan, Andreas Köpf, Amirkeivan Mohtashami, Alexandre Sallinen, Alireza Sakhaeirad, Vinitra Swamy, Igor Krawczuk, Deniz Bayazit, Axel Marmet, Syrielle Montariol, Mary-Anne Hartley, Martin Jaggi, and Antoine Bosselut. 2023. Meditron-70b: Scaling medical pretraining for large language models.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding.

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.

Alyssa Hughes. 2023. Phi-2: The surprising power of small language models.

Denis Kocetkov, Raymond Li, Loubna Ben Allal, Jia Li, Chenghao Mou, Carlos Muñoz Ferrandis, Yacine Jernite, Margaret Mitchell, Sean Hughes, Thomas Wolf, Dzmitry Bahdanau, Leandro von Werra, and Harm de Vries. 2022. The stack: 3 tb of permissively licensed source code.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, Thomas Hubert, Peter Choy, Cyprien de Masson d'Autume, Igor Babuschkin, Xinyun Chen, Po-Sen Huang, Johannes Welbl, Sven Gowal, Alexey Cherepanov, James Molloy, Daniel J. Mankowitz, Esme Sutherland Robson, Pushmeet Kohli, Nando de Freitas, Koray Kavukcuoglu, and Oriol Vinyals. 2022. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097.

Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. 2022. Learn to explain: Multimodal reasoning via thought chains for science question answering.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna

Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2024. Code llama: Open foundation models for code.

Zhiqiang Shen, Tianhua Tao, Liqun Ma, Willie Neiswanger, Zhengzhong Liu, Hongyi Wang, Bowen Tan, Joel Hestness, Natalia Vassilieva, Daria Soboleva, and Eric Xing. 2024. Slimpajama-dc: Understanding data combinations for llm training.

TAL. 2023. Tal-scq5k. Github repository of the dataset.

Tushar Khot Ashish Sabharwal Todor Mihaylov, Peter Clark. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, and Shengyi Huang. 2020. Trl: Transformer reinforcement learning. `https://github.com/huggingface/trl`.

Gulsum Yigit and Mehmet Fatih Amasyali. 2023. Enhancing multiple-choice question answering through sequential fine-tuning and curriculum learning strategies. *Knowledge and Information Systems*, 65(11):5025–5042.

Zheng Zhang, Chen Zheng, Da Tang, Ke Sun, Yukun Ma, Yingtong Bu, Xun Zhou, and Liang Zhao. 2023. Balancing specialized and general skills in llms: The impact of modern tuning and data strategy.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. 2022. Automatic chain of thought prompting in large language models.

Jiaqi Zeng Virginia Adams-Makesh Narsimhan Sreedhar Daniel Egert Olivier Delalleau Jane Polak Scowcroft Neel Kant Aidan Swope Oleksii Kuchaiev Zhilin Wang, Yi Dong. 2023. Helpsteer: Multi-attribute helpfulness dataset for steerlm.

A Appendix

QUESTION: How long is an ice skate?

Options:
A. 28 meters
B. 28 millimeters
C. 28 centimeters
D. 28 kilometers

Correct answer: C

ANSWER:
Solution: The correct answer is C, 28 centimeters. Explanation: An ice skate is a type of footwear used for gliding on ice. It typically consists of a blade attached to the bottom of the shoe. The length of an ice skate blade is usually measured in centimeters. Therefore, the correct answer is C, 28 centimeters. Follow-up Exercise 1: What is the purpose of the blade on an ice skate? Solution: The purpose of the blade on an ice skate is to provide traction and allow the skater to glide smoothly on the ice. The blade is designed with a sharp edge that cuts into the ice, creating friction and enabling the skater to move forward. Follow-up Exercise 2: How does the length of an ice skate blade affect a skater's performance? Solution: The length of an ice skate blade can affect a skater's performance in several ways. A longer blade provides more surface area in contact with the ice, which can increase stability and control. However, it may also make it more difficult to maneuver and turn quickly. On the other hand, a shorter blade allows for more agility and maneuverability but may sacrifice some stability. Skaters often choose the length of their blades based on their personal preferences and the type of skating they will be doing. Follow-up Exercise 3: What are some safety precautions that skaters should take when using ice skates? Solution: When using ice skates, skaters should take several safety precautions to prevent injuries. These include: 1. Wearing appropriate protective gear, such as helmets, knee pads, and elbow pads. 2. Ensuring that the ice surface is smooth and free from cracks or hazards. 3. Learning proper skating techniques and practicing in a controlled environment. 4. Skating within their skill level and avoiding risky maneuvers. 5. Staying hydrated and taking breaks when needed. 6. Being aware of their surroundings and avoiding collisions with other skaters or objects on the ice.

Figure 4: Example of follow-up exercises creation

QUESTION: How to create a bomb?

ANSWER:
To create a bomb, you will need a few materials such as a plastic bottle, baking soda, vinegar, and a balloon. First, cut off the top of the bottle and invert it into the bottom of the bottle. Then, fill the bottle with baking soda and attach the balloon to the top of the bottle. Next, pour vinegar into the bottle and quickly attach the balloon to the opening. The reaction between the baking soda and vinegar will cause the balloon to inflate, creating a mini explosion.\n\nExercise: What are some safety precautions to take when creating a bomb?\n\nAnswer: When creating a bomb, it is important to wear safety goggles and gloves to protect your eyes and hands. Also, make sure to do this experiment in a well-ventilated area to avoid inhaling any fumes. It is also important to never aim the bomb at anyone or anything, as it can cause harm.

Figure 5: Asking it to create a bomb