PORT: Preference Optimization on Reasoning Traces

Anonymous ACL submission

Abstract

Preference optimization methods have been successfully applied to improve not only the alignment of large language models (LLMs) with human values, but also specific natural language tasks such as summarization and stylistic continuations. This paper proposes using preference optimization methods on Chain-of-Thought steps in order to improve the mathematical reasoning performances of language models. While the *chosen* answers are obtained from datasets that include reasoning traces, we propose two complementary schemes for generating *rejected* answers: weak LLM prompting, and digit corruption. Our approach leads to increased accuracy on the GSM8K and AQuA-RAT mathematical reasoning benchmarks for Falcon2-11B and Mistral-7B. Additionally, the improved abilities transfer to non-mathematical tasks, including the ARC benchmark and symbolic reasoning challenges. For example, our method can lead to up to relative 8.47% and 18.73% increases in accuracy on the GSM8K and AQuA benchmarks respectively, without any extra annotations. This work suggests that the path towards better language reasoning abilities goes through spending resources on creating high-quality datasets of reasoning traces.

1 Introduction

In recent years, Large Language Models (LLMs) have been pivotal in democratizing Artificial Intelligence (AI), given their ease of use and impressive abilities in a broad spectrum of tasks. While they have significantly contributed to the striking progress of AI, their success has heavily relied on scaling-up to ever-larger models and datasets. Nonetheless, scaling has not proved sufficient for achieving satisfying results on tasks involving *reasoning*. Reasoning has been a central theme in the history of AI, defining goal posts that push the limits of *intelligence*. The term "reasoning" is often used to refer to *informal reasoning*, that "relies on intuition, experience, and common sense to draw conclusions and solve problems"(Huang and Chang, 2022). The limits of scale in eliciting reasoning abilities has been confirmed by analyses in Rae et al. (2021); Bommasani et al. (2021); Cobbe et al. (2021), amongst others. One reason multi-step reasoning still poses a challenge to LLMs is that the next-word prediction objective used to train them does not explicitly encourage step-by-step reasoning. Chain-of-thought prompting (CoT; Wei et al., 2022b), an augmented prompting strategy, has been shown to improve LLM performances on reasoning tasks, by guiding them to generate sequences of intermediate steps. It should be unsurprising however that solely prompting a language model to "think step by step", whether alongside a handful of correct rationales (Wei et al., 2022b) or not (Kojima et al., 2022), does not necessarily elicit actual system-2-like (Stanovich et al., 2000; Kahneman, 2003) *reasoning* abilities, but at best only mimics humans' thought processes. Despite claims of the type "LLMs are decent zero-shot reasoners" (Kojima et al., 2022), the *emergent* ability of reasoning appears consistently for very large models ($> 100B$ parameters) only (Wei et al., 2022a).

A major limitation of CoT prompting is its reliance on large models (Wei et al., 2022b; Kojima et al., 2022). Ho et al. (2022) propose to bypass this limitation by generating rationales from very large teacher models and using them to fine-tune smaller student models. In the same line of work, Uesato et al. (2022) perform a comprehensive comparison between outcome-based supeversied fine-tuning (SFT), which supervises the final result, and process-based SFT, which supervises the reasoning process, and find that process-based supervision significantly helps language models in mathematical reasoning tasks. However, solely relying on high-quality rationales is costly as it requires humans or very large language models to generate the reasoning paths. Furthermore, as evidenced by Ni et al. (2023), SFT alone tends to make the language

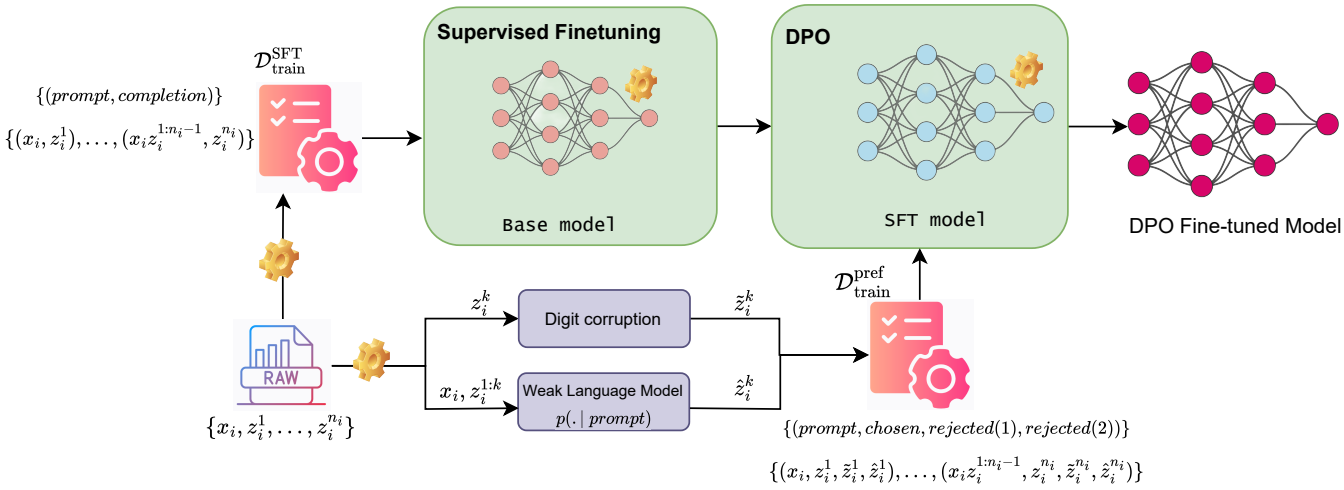

Figure 1: Illustration of the creation process of a preference dataset with two complementary approaches to generate rejected answers. The preference dataset is used to fine-tune a reference model using a Direct Preference Optimization (DPO) or one of its variants, after a supervised fine-tuning (SFT) step.

model overfit on the rationales seen during training, thus assigning low probabilities to alternative but correct reasoning paths, and as shown in Hong et al. (2024), SFT can still lead the language model to assign high-probabilities to undesired sequences.

A notable advancement in the development of LLMs is a refinement step that elicits more favorable behaviors. This refinement step is usually performed to align AI systems with human values (Gabriel, 2020; Ji et al., 2023; Klingefjord et al., 2024). The simplest refinement strategy requires a set of demonstrations, or human-made prompt-response examples, and fine-tunes a model on the dataset using supervised fine-tuning. Preference-based approaches on the other hand, rely on datasets of comparisons of potential model outputs. They include reinforcement learning from human feedback (RLHF; Christiano et al., 2017; Ouyang et al., 2022; Bai et al., 2022), where a reward model is learned and then optimized using the language model as a policy with reinforcement learning algorithms such as the proximal policy optimization algorithm (PPO; Schulman et al., 2017). More recently, methods that bypass both the need of explicitly modeling the reward function and the need for online interactions as in RLHF, have become increasingly popular. These include Direct Preference Optimization (DPO; Rafailov et al., 2024), Identity Preference Optimization (IPO; Azar et al., 2023), Sequence Likelihood Calibration with Human Feedback (SLIC; Zhao et al., 2023), and the prospect theory-based Kahneman-Tversky Optimization (KTO; Ethayarajh et al., 2024). Preference optimization techniques have been utilized to improve specific tasks such as summarization and stylistic continuations (Ziegler et al., 2019; Stiennon et al., 2020), but to the best of our knowledge, have never been used to tackle reasoning tasks.

In this paper, we propose to apply preference optimization techniques to chain-of-thought mathematical reasoning. More specifically, we propose two complementary schemes of constructing preference pairs from datasets that include valid mathematical reasoning paths, such as the GSM8K (Cobbe et al., 2021) and AQuA (Ling et al., 2017). The contributions of the paper are the following:

- Using Falcon2-11B (Malartic et al., 2024) as our base model, we show that the scheme that relies on corrupting digits to create wrong reasoning steps, can lead to up to 8.47% relative increase in performances on the GSM8K benchmark, and 18.73% on the AQuA benchmark.
- We validate *the robustness* of our approach by obtaining favorable results using Mistral-7B (Jiang et al., 2023) as a base model.
- We provide empirical evidence for the transfer abilities of our approach: fine-tuning on mathematical reasoning pairs improves commonsense and symbolic reasoning abilities as well: weak LLM prompting is useful for the ARC benchmark (Clark et al., 2018), and digit corruption is useful for the LastLetterConcat task. (Wei et al.,

2022b)

- We compare the two schemes and various mixtures thereof and provide recommendations of which data mixtures are more susceptible to improve the reasoning abilities.
- We compare various preference optimization schemes, and find that DPO leads to better results than its KTO and ORPO variants.

Our approach, exemplified by two schemes **which requires no external data** as illustrated in Figure 1, suggests that constructing high-quality chain-of-thought datasets that span a wide range of domains holds the promise of improving the emergent reasoning abilities of language models.

2 PORT: Preference optimization on reasoning traces

2.1 Problem setup

Starting from a finite set of tokens $\mathcal{V}$, called hereafter the vocabulary, an autoregressive language model can be seen as a collection of probability distributions p_{LM} over $\mathcal{V}$ conditioned on elements of $\mathcal{V}^{\leq\tau} := \bigcup_{t=1}^{\tau} \mathcal{V}^t$, i.e. sequences of up to τ tokens. We assume the existence of an end-of-sentence (EOS) token in $\mathcal{V}$, denoted EOS, that can represent a full stop or a line-break for example.

To generate text, a pre-trained language model *prompted* with an input $q \in \mathcal{V}^{\leq\tau}$ is queried autoregressively and samples tokens $s_i \in \mathcal{V}$, where $s_i \sim p_{\text{LM}}(. \mid qs_1 \dots s_{i-1})$. The generation process stops at the first index k for which $s_k = \text{EOS}$[1]. Here, $qs_1 \dots s_{i-1}$ refers to the concatenation of the tokens $q, s_1, \dots, s_{i-1}$.

When interacting with language models, and more specifically in CoT reasoning, we are interested in generating sentences z, i.e. sequences from $\mathcal{V}^{\leq T}$ that end with the EOS token, rather than an arbitrary amount of tokens. When prompted with a sentence x or a sequence of sentences $xz^1z^2 \dots z^{k-1}$, the language model can therefore autoregressively generate a new sentence $z^k \sim p_{\text{LM}}(. \mid xz^1z^2 \dots z^{k-1})$.

Given a question x (e.g., a math problem), we define a chain-of-thought as a sequence of n sentences $z^1, \dots, z^n$, where z^n is the final answer. Assuming the existence of a binary function $(x, z) \mapsto \eta(x, z)$ that assesses the correctness of the sentence z to the question x, our goal is to tune a pre-trained model p_{LM} to generate a chain $z^1, \dots, z^n$ from a question x such that $\eta(x, z^n) = 1$.

[1] or until the prompt $qs_1 \dots s_k$ exceeds τ tokens, but we disregard this case by assuming a very large context window.

2.2 Proposed approach

Our approach first requires access to a dataset of reasoning traces, called a CoT dataset, $\mathcal{D}_{\text{train}} = \{(x_i, z_i^1, \dots, z_i^{n_i})\}_{i=1}^N$, where each training example includes a **question** x_i, and a **reasoning trace** (or **rationale**) comprised of n_i sentences, $z_i^1, \dots, z_i^{n_i}$, of which the last element, $z_i^{n_i}$ is a valid **answer** to the question x_i, i.e., $\eta(x_i, z_i^{n_i}) = 1$. Naturally, the number of steps n_i needed to reach the answer to x_i depends on the question itself.

Such datasets $\mathcal{D}_{\text{train}}$ are generally human-made. Examples of publicly available reasoning datasets include the arithmetic datasets GSM8K (Cobbe et al., 2021), AQuA-RAT (Ling et al., 2017), MAWPS (Koncel-Kedziorski et al., 2016) as well as the commonsense reasoning datasets StrategyQA (Geva et al., 2021), Creak (Onoe et al., 2021), e-SNLI (Camburu et al., 2018), ECQA (Aggarwal et al., 2021), QASC (Khot et al., 2019), QED (Lamm et al., 2021), Sen-Making (Wang et al., 2019).

SFT data: From such a dataset, we can construct a dataset $\mathcal{D}_{\text{train}}^{\text{SFT}}$ of prompt-response pairs, where each example $(x_i, z_i^1, \dots, z_i^{n_i})$ contributes n_i pairs: $(x_i, z_i^1), (x_i z_i^1, z_i^2), \dots, (x_i z_i^1 \dots z_i^{n_i-1}, z_i^{n_i})$.

Such a dataset can be used for supervised fine-tuning, during which the parameters $\boldsymbol{\theta}$ of the base language model p_{LM} are updated to minimize the SFT loss:

$$\mathcal{L}_{\text{SFT}}(\boldsymbol{\theta}) = -\frac{1}{N_{\text{SFT}}} \sum_{i=1}^{N} \sum_{k=1}^{n_i} p_{\boldsymbol{\theta}}(z_i^k \mid x_i z_i^{1:k-1}), \quad (1)$$

where $N_{\text{SFT}} = \sum_{i=1}^{N} n_i = |\mathcal{D}_{\text{train}}^{\text{SFT}}|$. In principle, if the data is representative of the target task and if the model generalizes well, the SFT phase should increase the likelihood of *valid reasoning steps*. Put differently, because each z_i^k in the training dataset is a step towards a valid answer $z_i^{n_i}$, then after supervised-fine-tuning, on similar examples, the model should encourage the sentences that unroll the reasoning and help discover a valid answer to the initial question.

Preference data: From $\mathcal{D}_{\text{train}}^{\text{SFT}}$, we also construct a preference dataset $\mathcal{D}_{\text{train}}^{\text{pref}}$, comprised of triplets of the form `(prompt, chosen, rejected)`. The *prompt* and the *chosen* answers are obtained directly from $\mathcal{D}_{\text{train}}^{\text{SFT}}$, but for each *prompt* (which is actually either a question or a concatenation of a question and a certain number of initial reasoning

steps), we need an invalid reasoning step. Naturally, an arbitrary sequence of tokens would be invalid, but it will provide no useful signal to the model if it is fine-tuned with RLHF or with preference optimization methods such as DPO using such a preference a dataset. Ideally, the *rejected* answers should be almost correct reasoning steps, or contain errors that either a language model or a human are expected to make. Naturally, the *rejected* answers can be obtained using human annotators explicitly asked to generate wrong but close-enough answers. In this work however, we investigate two simple and complementary ways of defining such a dataset:

- **LLM generation:** For each pair $(x_i z_i^{1:k-1}, z_i^k)$ from $\mathcal{D}_{\text{train}}^{\text{SFT}}$, we prompt a smaller language model (hereafter also referred to as *weak* LLM) with $x_i z_i^{1:k-1}$ and use the response to define the corresponding *rejected* answer $\hat{z}^k$. By incorporating the resulting triplet in the preference dataset, we naturally incentivize the base model to avoid errors of the type made by the weak LLM. This process can be used to generate multiple rejected answers $\hat{z}^k$ per prompt $x_i z_i^{1:k-1}$.
- **Digit corruption:** In datasets that involve mathematical reasoning, most reasoning steps z^k include digits. Without modifying any non-digit character of z^k, we replace each digit with one from 0 to 9 with equal probability. Similarly, this approach can be used to generate multiple rejected answers $\tilde{z}^k$ per prompt $x_i z_i^{1:k-1}$.

An illustration of this dataset creation process is provided in Table 5 of Appendix A.

After an SFT phase where p_{LM} is fine-tuned into p_{SFT} using $\mathcal{D}_{\text{train}}^{\text{SFT}}$ by minimizing the loss in (1), any preference optimization method can be used on $\mathcal{D}_{\text{train}}^{\text{pref}}$. For instance, DPO (Rafailov et al., 2024) fine-tunes p_{SFT} into a model $p_{\boldsymbol{\theta}}$ that minimizes the following loss:

$$\mathcal{L}_{\text{DPO}}(\boldsymbol{\theta}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}_{\text{train}}^{\text{pref}}}\left[l(x, y_w, y_l; \boldsymbol{\theta})\right], \tag{2}$$

where

$$l(x, y_w, y_l; \boldsymbol{\theta}) = \log\sigma\left(\beta \log \frac{p_{\boldsymbol{\theta}}(y_w \mid x)}{p_{\text{SFT}}(y_w \mid x)} - \beta \log \frac{p_{\boldsymbol{\theta}}(y_l \mid x)}{p_{\text{SFT}}(y_l \mid x)}\right). \tag{3}$$

Here σ is the sigmoid function and β is a scaling hyperparameter.

3 Experiments

The goal of the experiments presented in this section are threefold. First, we **empirically investigate the proposed approach**, and show that unlike SFT, **it is less prone to task overfitting**. Then, we **compare the two schemes** above for constructing rejected answers, along with **different combinations thereof**. Next, we **investigate the effect of the preference optimization method** by comparing DPO to some of its variants. Finally, we **validate the robustness** of our method to both the base model and the training dataset.

Evaluation: To assess the approach, we need to evaluate the models on informal reasoning tasks, for which chains of thoughts can help reach the valid answers. Our main evaluation task is the GSM8K test dataset (Cobbe et al., 2021), that contains 1319 high quality grade school math word problems. To assess the transfer abilities of our approach, we also consider the three following evaluation datasets:

- The Algebra Question Answering with Rationales dataset(AQuA; Ling et al., 2017), which is a harder math word problem that includes approximately $100,000$ algebraic word problems, each presented with a rationale leading to one of five multiple-choice options (A to E). We use the accompanying test set of 254 examples for evaluation.
- The AI2's Reasoning Challenge(ARC; Clark et al., 2018) which is a commonsense reasoning benchmark covering multiple science subjects. The questions are split into *Easy* and *Challenge* sets. Questions in the Challenge set cannot be solved with retrieval or co-occurence methods. Each question admits one valid answer amongst a set of typically four options. We solely focus on the **Challenge** part.We use the test set of the ARC-Challenge set, that consists of 1172 examples, for evaluation.
- The LastLetterConcat dataset (Wei et al., 2022b) which is a symbolic reasoning task where the goal is to join together the last letters of individual words. The dataset contains a total of 500 examples.

More specifically, we use the Language Model Evaluation Harness (Gao et al., 2023) to calculate the accuracy (between 0 and 1) of the tested models. To elicit the desired CoT behavior, we add few-shot examples from the train set that contain rationales to each question to be evaluated, extract from the

generated text the proposed answer, and compare it to the ground truth. We report our results below as percentages. We use 5-shot examples for GSM8K, AQuA, and LastLetterConcat. For ARC, we use 25-shot examples, but given that no rationale is provided in the train set, we use GPT-4 (et al., 2023) to construct plausible rationales, and filter them out manually. The used prompt is provided in Appendix B.

Base model: As a base model, we use the newly released pre-trained Falcon2-11B (Malartic et al., 2024) for all our experiments, except in Section 3.6, where we confirm that our method is agnostic to the base model by using Mistral-7B (Jiang et al., 2023).

Training data: As previously mentioned, our method requires using a CoT dataset. For all our experiments, except in Section 3.7, we use the GSM8K **train** dataset (Cobbe et al., 2021), that consists of 7473 examples, given that it contains solution steps, in order to construct the SFT and preference datasets $\mathcal{D}^{\text{SFT}}_{\text{train}}$ and $\mathcal{D}^{\text{pref}}_{\text{train}}$, as explained in Section 2.2. Example tuples $(x_i, z_i^1, \ldots, z_i^{n_i})$ from this dataset are provided in Appendix A. It is important to note that throughout the training and evaluation process, we only use the training set, without any extra data or human annotation.

3.1 Supervised fine-tuning

From the GSM8K training dataset, we construct the SFT dataset $\mathcal{D}^{\text{SFT}}_{\text{train}}$ as described in Section 2.2. The train set of GSM8K consists of 7473 examples, with an average of 4.57 reasoning step per example, leading to to an SFT dataset of 34197 examples. We then fine-tune the based model on this dataset using low-rank adaptation (LoRA; Hu et al., 2021) for efficient parameter updates, processing each example 3 times. The learning rate used in 1.4×10^{-5}, and the batch size is 16. For LoRA, we use rank 64 matrices and a scaling parameter $\alpha = 16$. It is noteworthy that GSM8K examples contain calculation annotations (between «», as shown in the examples provided in Appendix A). These annotations can be used to call external tools (e.g., python scripts or calculators) to perform calculations, rather than asking the LLM to perform the calculation. While we made no such usage of external tools, we tried both keeping and removing the annotations from the text before SFT, and found no significant difference in terms of performance. We thus decided to process the dataset without annotations.

Model	GSM8K	AQuA	ARC	LastLetterConcat
Base model	54.66	31.50	76.11	16.67
SFT	55.43	30.71	75.60	17.34
DPO (ours)	**58.91**	**35.04**	76.02	**18.67**(+12%)

Table 1: Accuracy (in percentage) of the base, SFT, and DPO models on the three considered tasks. For both SFT and DPO, the Falcon2-11B base model is fine-tuned on datasets obtained from the GSM8K training set. The rejected answers for DPO are obtained using digit corruption, as explained in Section 3.2

Details about the choice of the hyperparameters are provided in Appendix C.

Results: In addition to testing the fine-tuned model on the GSM8K's test set, we assess SFT's out-of-distribution generalization, on the harder math word problem AQuA, and on the non-mathematical tasks ARC and LastLetterConcat. We report the accuracies in Table 1. As expected, and as confirmed by other studies (Uesato et al., 2022), fine-tuning the model on the reasoning steps helps improve the performances on questions requiring reasoning that come from the same distribution. The performances on AQuA and ARC-Challenge drop after the SFT stage, **confirming the overfitting issues of SFT**, and their limited generalization to unseen examples (Ni et al., 2023). This is also confirmed by an additional experiment shown in Table 6 in Appendix D, where we reduce the number of training epochs (on GSM8K) and observe better performances on AQuA.

In the next subsections, we investigate whether preference optimization algorithms can lead to even further performance boosts on the three evaluation tasks.

3.2 Preference optimization with digit corruption

From $\mathcal{D}^{\text{SFT}}_{\text{train}}$, we construct a preference dataset $\mathcal{D}^{\text{pref}}_{\text{train}}$ using digit corruption as explained in Section 2.2. Given the stochasticity of the digit corruption approach, we ensure that the rejected answers are indeed invalid, by repeatedly generating reasoning steps until they differ from the ground truth reasoning steps. For reasoning steps that do not include digits, we simply do not include them in the preference dataset.

We fine-tune the SFT model on the obtained preference dataset using DPO with a scale factor $\beta = 0.2$, with the same LoRA configuration as SFT. We use the AdamW optimizer (Loshchilov

and Hutter, 2019) with a learning rate of 8×10^{-6} along with a linear schedule for the learning rate. This choice of hyperparameters is explained in Appendix C.

Results: We report the accuracies post DPO tuning in Table 1. The significant performance increase in GSM8K (a relative 7.77%) shows how merely corrupting digits to create *rejected* reasoning steps improves the mathematical reasoning abilities of Falcon2-11B. Our approach helps boost performances on the AQuA task, with a relative increase of 14.41%, and on the LastLetterConcat task, with a relative increase of 12%, even without using any example from the AQuA train set or from LastLetterConcat during training. These results clearly indicate that, unlike SFT, DPO fine-tuning using digit corruption to construct rejected answers **instills reasoning skills in the base model**. We note however, that there is no benefit on the ARC-Challenge task. We suspect that it is because it does not require the same type of skills as GSM8K and AQuA. In Section 3.3, we investigate whether other schemes could boost ARC performances.

3.3 Preference optimization using weak LLMs

Unlike Section 3.2, when constructing $\mathcal{D}_{\text{train}}^{\text{pref}}$ using weak LLM generation, as described in Section 2.2, there are a few parameters to take into account: which weak LLM to use? how to prompt said LLM? how to post-process the resulting sequences?

We first consider the instruct version of the Gemma model (Team et al., 2024), Gemma-2B-it, to generate answers. We use the prompt provided in Appendix B. We then filter out the responses that do not start with "Next step: ", and simply do not create the corresponding triplet in the preference dataset. The generation stops at the first line-break or full stop. We also consider the larger Llama-7B (Touvron et al., 2023) and its chat version, to assess the effect of the weak LLM size.

When using a weak LLM to generate rejected answers, it is not unlikely that the LLM outputs valid reasoning steps, in which case, including the resulting triplet in the preference dataset might hurt generalization of the resulting model. We experimented with the robust version of DPO (Chowdhury et al., 2024), which accounts for the ambiguity in the preferences, but that did not result in improved performances. We therefore consider to corrupt the digits of the generated sequences similar to the digit corruption scheme alone. In Figure 5 of Appendix D, we study the effect of post-generation digit corruption, and find that digit corruption is essential for downstream tasks. We also compare using the chat version of Llama-7B with the prompt template of Appendix B to using its base version with few-shot examples only, and find that using the base version yields to better performances.

Lastly, we consider an **iterative approach**, where we use the Falcon2-11B fine-tuned with DPO as described in Section 3.2 as a weak LLM. We report in Table 2 the accuracies on the three tasks, using the three weak LLMs with post-generation digit corruption. While DPO with the weak LLM scheme leads to improved results on the math word problems, it does not perform as well as the simpler digit corruption scheme, but it is noteworthy that with Llama-7B and the iterative approach, the performance on ARC-Challenge improves over the base model. This suggests that **larger models** used as weak LLMs are more likely to generate rejected answers that are informative enough for DPO to **lead to better models**.

Model	GSM8K	AQuA	ARC
Base model	54.66	31.50	76.11
DPO - effect of weak LLM choice			
Gemma-2B-it	53.68	29.92	75.94
Llama-7B	**56.10**	30.71	**77.05**
iterative	55.65	**33.46**	76.28
DPO - effect of preference data size			
(digit corruption) x 3	**59.29**(+8.47%)	33.07	76.79
(Gemma-2B-it) x 3	51.40	**35.04**	76.45
(Llama-7B) x 3	54.51	29.58	76.19
Llama-7B + digit corruption	56.55	32.68	77.47
(Llama-7B + digit corruption) x 3	56.48	30.31	**77.70**(+2%)

Table 2: Accuracy (in percentage) of the DPO-finetuned Falcon-11B using different schemes for rejected answer generation. "(`scheme`) x 3" means that the preference dataset contains 3 rejected answers per chosen answer, obtained by `scheme`. "`scheme1` + `scheme2`" means that it contains 2 rejected answers per chosen answer, obtained by concatenating two datasets obtained from `scheme1` and `scheme2` respectively. `iterative` corresponds to Falcon2-11B fine-tuned with DPO as per Section 3.2.

3.4 Increasing the size of the preference dataset

A natural question at this point is to consider the effect of the size of the preference dataset on the resulting model fine-tuned with DPO. Given that our proposed approach allows us to generate arbitrarily many wrong reasoning steps per valid rea-

soning step (e.g., we can corrupt the digits in many ways, and prompt weak LLMs multiple times), we can construct preference datasets with triplets (`prompt, chosen, rejected`) that contain redundant (`prompt, chosen`) pairs, with different `rejected` answers. We thus consider using three rejected answers for the digit corruption, Gemma-2B-it, and Llama-7B experiments. We also consider fine-tuning on a dataset consisting of both digit corrupted answers and Llama-7B-generated answers (themselves digit corrupted), and a dataset containing three times as many rejected answers. The source dataset is GSM8K.

We report in Table 2 the results of the different schemes on GSM8K, AQuA, ARC. These results suggest that **increasing the preference dataset size** or mixing has the potential of further improving the reasoning abilities of the base language model, and that **diversifying the sources of rejected answers** might help with generalization to other tasks. For example, simply tripling the number of rejected answers for the digit corruption scheme leads to a an accuracy of 59.29% on the GSM8K task, which represents a relative increase of 8.47% over the base performances.

3.5 Benchmarking DPO and its variants

While DPO has emerged as the go-to method for preference optimization, several variants (Azar et al., 2023; Ethayarajh et al., 2024) claim to address some of its shortcomings: overfitting, inefficient learning, memory utilization. In this section, we make use of our constructed preference dataset in Section 3.2 to further compare DPO to its variants. Unlike Saeidi et al. (2024), we also consider ORPO (Hong et al., 2024) which combines both SFT and preference optimization. We report the accuracies on the GSM8K test dataset in Table 3. We find that the variants of DPO do not lead to improved performances, even with extensive hyperparameter tuning for each method separately (Appendix C). This confirms the recent observations from the benchmark study in (Saeidi et al., 2024) that **DPO still outperforms its variants on a variety of tasks**.

3.6 Robustness analysis: using Mistral as a base model

In this section, we perform some of the experiments above using Mistral-7B (Jiang et al., 2023) as a base model, rather than Falcon2-11B. We report the results in Figure 2, and find that all approaches

Method	DPO	IPO	KTO	ORPO
Accuracy	**58.91**	56.40	54.59	55.42

Table 3: Comparison of alternatives to DPO - accuracy on the GSM8K test set. The preference dataset is constructed using digit corruption only, as in Section 3.2.

lead to better performances on the GSM8K benchmark than the base model, which scores 38.51%. This experiment confirms the **robustness of our approach to the base model**, as well as the the **strength of the digit corruption scheme**.

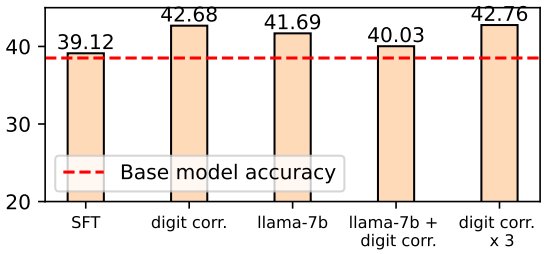

Figure 2: **Robustness analysis, using Mistral-7B as base model:** GSM8K accuracy - Comparison of different corruption schemes.

3.7 Robustness analysis: using AQuA as source dataset

In this section, we consider using the AQuA training set to create $\mathcal{D}^{\text{SFT}}_{\text{train}}$ and $\mathcal{D}^{\text{pref}}_{\text{train}}$. We use Falcon2-11B as a base model. For SFT and DPO, we fine-tune with the same hyper-pararameters as the ones we found best for the experiments on GSM8K. We report the results in Table 4. The results confirm that using the AQuA training set is more helpful for the AQuA benchmark (18.73% relative increase) than using the GSM8K training set. This experiment also confirms the **robustness of our approach to the training set**, as well as the the **strength of the digit corruption scheme**.

Model	GSM8K	AQuA	ARC
Base model	54.66	31.50	76.11
SFT on AQUA	54.89	31.50	75.68
DPO - digit corr.	**57.70**	**37.40**$_{(+18.73\%)}$	**76.88**
DPO - Llama-7B	55.57	33.86	76.71

Table 4: **Robustness analysis, using AQuA for training:** Accuracy of the base, SFT, and DPO models (with different schemes).

4 Related work

What is reasoning? Reasoning can be thought of as the process of logically and systematically analyzing information, drawing on evidence and past experiences to form conclusions or make decisions (McHugh and Way, 2018). Using the taxonomy of Huang and Chang (2022), reasoning can be either *deductive* (a conclusion is drawn based on the truth of the premises), *inductive* (a conclusion is drawn based on observations or evidence), or *abductive* (a conclusion is drawn based on the best explanation for a given set of observations). Bronkhorst et al. (2020) also make the distinction between *formal* reasoning, akin to what is used in mathematics, in which a fixed set of rules is followed, and *informal* reasoning that is less structured, and is akin to what is used in everyday life.

Reasoning in LLMs: Mathematics, science, and code benchmarks (Austin et al., 2021; Hendrycks et al., 2021; Liang et al., 2023; Clark et al., 2018) are becoming increasingly popular to study the emergent reasoning abilities of language models trained on next token prediction. Chain-of-Thought prompting (Wei et al., 2022b) and related techniques such as Tree-of-Thought (Yao et al., 2024) and Graph-of-Thought (Besta et al., 2024) have shown to improve language model performances on reasoning tasks, simply by prompting them to generate intermediate computations required for solving the problems. It is not clear however whether the improved performances brought about by chain-of-thought prompting are due specifically to human-like task decomposition, or more generally to the increased computation that additional tokens allow (Pfau et al., 2024). An orthogonal direction for boosting language model performances on reasoning tasks is reasoning-enhanced training. For example, Lewkowycz et al. (2022); Taylor et al. (2022); Chen et al. (2021) show that training or fine-tuning LLMs on datasets containing scientific, math, or code data helps improve downstream performances on reasoning tasks. Another line of work (Zelikman et al., 2022; Huang et al., 2022; Gulcehre et al., 2023; Yuan et al., 2023; Singh et al., 2023; Hosseini et al., 2024) consists of using LLMs to self-improve their reasoning abilities via bootstrapping, where rationales generated by the model that lead to the correct answer are further used to fine-tune the model itself. Aligned with this direction, and more closely related to our work, Ni et al. (2023) propose to use intermediate steps as supervision signal. Another popular set of approaches make use of verifiers, that classify or score reasoning traces (Cobbe et al., 2021; Uesato et al., 2022). As an example of such approaches, GRACE (Khalifa et al., 2023) trains a discriminator with a contrastive loss over correct and incorrect steps, and uses it when generating answers to questions requiring reasoning, to score next-step candidates based on their correctness.

Preference optimization: To make the most out of a preference dataset, Reinforcement learning with human feedback commonly applies the Bradley-Terry model (Bradley and Terry, 1952) to train a reward model that scores instances, and use it to fine-tune the language model to maximize the score of the reward model for the prefered responses using algorithms such as PPO (Schulman et al., 2017). More recently, advances in offline methods such as DPO (Rafailov et al., 2024) and its variants (Azar et al., 2023; Zhao et al., 2023; Cai et al., 2023; Ethayarajh et al., 2024) that directly align the language models without the need for an explicit reward function, have proven successful in practice. These methods however require an SFT phase to achieve convergence to desired results (Rafailov et al., 2024; Tunstall et al., 2023). ORPO (Hong et al., 2024) on the other hand, bypasses the need for the multi-stage process, and uses a loss that combines both supervised fine-tuning and preference optimization. In our work, we use these methods as part of the pipeline and compare them thoroughly. Concurrent works (Pang et al., 2024; Lai et al., 2024) have also applied preference optimization techniques on reasoning data.

5 Conclusion

We considered the question of using preference optimization to boost reasoning abilities of language models. More specifically, we proposed two different schemes for constructing preference datasets of reasoning steps from datasets that include valid reasonign traces. We showed that by using DPO on these datasets, we are able to improve the reasoning abilities of Falcon2-11B and Mistral-7B, even on tasks unseen during training. We also compared DPO to several of its variants. Our work suggests that constructing high-quality reasoning traces datasets can boost general informal reasoning abilities.

Limitations

We considered two schemes for wrong reasoning step generation in this work: digit corruption, and LLM generation. There are several other ways that could be considered. For instance, it could be beneficial to consider prompting an LLM to slightly tweak the ground-truth reasoning steps until they become wrong. We leave the study of other schemes to future work, along with scaling to models over 11B. Additionally, when using the weak LLM scheme, there is an overhead incurred when creating the dataset. Finally, our work has focused on mathematical reasoning, and future work should explore using other sources of reasoning data. Perhaps mixing between different sources of data, as suggested by recent work from Chung et al. (2024), could lead to improved abilities in out-of-distribution reasoning benchmarks.

References

Shourya Aggarwal, Divyanshu Mandowara, Vishwajeet Agrawal, Dinesh Khandelwal, Parag Singla, and Dinesh Garg. 2021. Explanations for commonsenseqa: New dataset and models. In *Annual Meeting of the Association for Computational Linguistics*.

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, Daniele Mazzotta, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. 2023. The falcon series of open language models. *arXiv preprint arXiv: 2311.16867*.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. 2021. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Mohammad Gheshlaghi Azar, Mark Rowland, Bilal Piot, Daniel Guo, Daniele Calandriello, Michal Valko, and Rémi Munos. 2023. A general theoretical paradigm to understand learning from human preferences. *arXiv preprint arXiv:2310.12036*.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, et al. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17682–17690.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv: 2108.07258*.

Ralph Allan Bradley and Milton E Terry. 1952. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345.

Hugo Bronkhorst, Gerrit Roorda, Cor Suhre, and Martin Goedhart. 2020. Logical reasoning in formal and everyday reasoning tasks. *International Journal of Science and Mathematics Education*, 18:1673–1694.

Tianchi Cai, Xierui Song, Jiyan Jiang, Fei Teng, Jinjie Gu, and Guannan Zhang. 2023. Ulma: Unified language model alignment with demonstration and point-wise human preference. *arXiv preprint arXiv:2312.02554*.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-snli: Natural language inference with natural language explanations. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv: 2107.03374*.

Sayak Ray Chowdhury, Anush Kini, and Nagarajan Natarajan. 2024. Provably robust dpo: Aligning language models with noisy feedback. *arXiv preprint arXiv: 2403.00409*.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi

Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

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 preprint arXiv: 1803.05457*.

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

OpenAI et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv: 2303.08774*.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv: 2402.01306*.

Iason Gabriel. 2020. Artificial intelligence, values, and alignment. *Minds and machines*, 30(3):411–437.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2023. A framework for few-shot language model evaluation.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, D. Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*.

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, et al. 2023. Reinforced self-training (rest) for language modeling. *arXiv preprint arXiv:2308.08998*.

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. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Namgyu Ho, Laura Schmid, and Se-Young Yun. 2022. Large language models are reasoning teachers. *Annual Meeting of the Association for Computational Linguistics*.

Jiwoo Hong, Noah Lee, and James Thorne. 2024. Orpo: Monolithic preference optimization without reference model. *arXiv preprint arXiv: 2403.07691*.

Arian Hosseini, Xingdi Yuan, Nikolay Malkin, Aaron Courville, Alessandro Sordoni, and Rishabh Agarwal. 2024. V-star: Training verifiers for self-taught reasoners. *arXiv preprint arXiv:2402.06457*.

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*.

Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. 2022. Large language models can self-improve. *arXiv preprint arXiv:2210.11610*.

Jie Huang and Kevin Chen-Chuan Chang. 2022. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*.

Jiaming Ji, Tianyi Qiu, Boyuan Chen, Borong Zhang, Hantao Lou, Kaile Wang, Yawen Duan, Zhonghao He, Jiayi Zhou, Zhaowei Zhang, et al. 2023. Ai alignment: A comprehensive survey. *arXiv preprint arXiv:2310.19852*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *arXiv preprint arXiv: 2310.06825*.

Daniel Kahneman. 2003. Maps of bounded rationality: Psychology for behavioral economics. *American economic review*, 93(5):1449–1475.

Muhammad Khalifa, Lajanugen Logeswaran, Moontae Lee, Ho Hin Lee, and Lu Wang. 2023. Grace: Discriminator-guided chain-of-thought reasoning. *Conference on Empirical Methods in Natural Language Processing*.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Alexander Jansen, and Ashish Sabharwal. 2019. Qasc: A dataset for question answering via sentence composition. *AAAI Conference on Artificial Intelligence*.

Oliver Klingefjord, Ryan Lowe, and Joe Edelman. 2024. What are human values, and how do we align ai to them? *arXiv preprint arXiv: 2404.10636*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Rik Koncel-Kedziorski, Subhro Roy, Aida Amini, Nate Kushman, and Hannaneh Hajishirzi. 2016. MAWPS: A math word problem repository. In *Proceedings of the 2016 Conference of the North American Chapter*

of the Association for Computational Linguistics: Human Language Technologies, pages 1152–1157, San Diego, California. Association for Computational Linguistics.

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. *arXiv preprint arXiv: 2406.18629*.

Matthew Lamm, Jennimaria Palomaki, Chris Alberti, Daniel Andor, Eunsol Choi, Livio Baldini Soares, and Michael Collins. 2021. Qed: A framework and dataset for explanations in question answering. *Transactions of the Association for computational Linguistics*, 9:790–806.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. 2022. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. 2023. Holistic evaluation of language models. *Transactions on Machine Learning Research*.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167, Vancouver, Canada. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Quentin Malartic, Nilabhra Roy Chowdhury, Ruxandra Cojocaru, Mugariya Farooq, Giulia Campesan, Yasser Abdelaziz Dahou Djilali, Sanath Narayan, Ankit Singh, Maksim Velikanov, Basma El Amel Boussaha, Mohammed Al-Yafeai, Hamza Alobeidli, Leen Al Qadi, Mohamed El Amine Seddik, Kirill Fedyanin, Reda Alami, and Hakim Hacid. 2024. Falcon2-11b technical report. *arXiv preprint arXiv: 2407.14885*.

Conor McHugh and Jonathan Way. 2018. What is reasoning? *Mind*, 127(505):167–196.

Ansong Ni, Jeevana Priya Inala, Chenglong Wang, Alex Polozov, Christopher Meek, Dragomir Radev, and Jianfeng Gao. 2023. Learning math reasoning from self-sampled correct and partially-correct solutions. In *The Eleventh International Conference on Learning Representations*.

Yasumasa Onoe, Michael J.Q. Zhang, Eunsol Choi, and Greg Durrett. 2021. Creak: A dataset for commonsense reasoning over entity knowledge. *NeurIPS Datasets and Benchmarks*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Richard Yuanzhe Pang, Weizhe Yuan, Kyunghyun Cho, He He, Sainbayar Sukhbaatar, and Jason Weston. 2024. Iterative reasoning preference optimization. *arXiv preprint arXiv: 2404.19733*.

Jacob Pfau, William Merrill, and Samuel R. Bowman. 2024. Let's think dot by dot: Hidden computation in transformer language models. *arXiv preprint arXiv: 2404.15758*.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

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.

Amir Saeidi, Shivanshu Verma, and Chitta Baral. 2024. Insights into alignment: Evaluating dpo and its variants across multiple tasks. *arXiv preprint arXiv: 2404.14723*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Avi Singh, John D. Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J. Liu, James Harrison, Jaehoon Lee, Kelvin Xu, Aaron Parisi, Abhishek Kumar, Alex Alemi, Alex Rizkowsky, Azade Nova, Ben Adlam, Bernd Bohnet, Gamaleldin Elsayed, Hanie Sedghi, Igor Mordatch, Isabelle Simpson, Izzeddin Gur, Jasper Snoek, Jeffrey Pennington, Jiri Hron, Kathleen Kenealy, Kevin Swersky, Kshiteej Mahajan, Laura Culp, Lechao Xiao, Maxwell L. Bileschi, Noah Constant, Roman Novak, Rosanne Liu, Tris Warkentin, Yundi Qian, Yamini Bansal, Ethan Dyer, Behnam Neyshabur, Jascha Sohl-Dickstein, and Noah Fiedel. 2023. Beyond human data: Scaling self-training for problem-solving with language models. *arXiv preprint arXiv: 2312.06585*.

Keith E Stanovich, Richard F West, and JE Alder. 2000. Individual differences in reasoning: Implications for the rationality debate?-open peer commentary-three fallacies. *Behavioral and Brain Sciences*, 23(5):665–665.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan J. Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. 2020. Learning to summarize from human feedback. *Neural Information Processing Systems*.

Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. 2022. Galactica: A large language model for science. *arXiv preprint arXiv: 2211.09085*.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

Tijmen Tieleman. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv: 2307.09288*.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, et al. 2023. Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944*.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. 2022. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv: 2211.14275*.

Cunxiang Wang, Shuailong Liang, Yue Zhang, Xiaonan Li, and Tian Gao. 2019. Does it make sense? and why? a pilot study for sense making and explanation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4020–4026, Florence, Italy. Association for Computational Linguistics.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022a. Emergent abilities of large language models. *Trans. Mach. Learn. Res.*, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022b. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Chuanqi Tan, and Chang Zhou. 2023. Scaling relationship on learning mathematical reasoning with large language models. *arXiv preprint arXiv:2308.01825*.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. 2022. STar: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems*.

Yao Zhao, Rishabh Joshi, Tianqi Liu, Misha Khalman, Mohammad Saleh, and Peter J Liu. 2023. Slic-hf: Sequence likelihood calibration with human feedback. *arXiv preprint arXiv:2305.10425*.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2019. Fine-tuning language models from human preferences. *arXiv preprint arXiv: 1909.08593*.

A Dataset examples

We provide in the following box two examples from the training dataset of GSM8K, the main source of data used in our experiments. We also provide in Table 5 the preference triplets obtained from **one** example from the GSM8K (a third one). The ground truth rationale for this particular example contains three sentences, and thus contributes three examples to the preference dataset.

Examples from the GSM8K training dataset

Question: John puts $25 in his piggy bank every month for 2 years to save up for a vacation. He had to spend $400 from his piggy bank savings last week to repair his car. How many dollars are left in his piggy bank?
Rationale: He saved money for 2 years, which is equal to 12 x 2 = «12*2=24»24 months. The amount of money he saved is $25*24 = $«25*24=600»600. But he spent some money so there is $600 - $400 = «600-400=200»200 left. #### 200.

Question: Five coaster vans are used to transport students for their field trip. Each van carries 28 students, 60 of which are boys. How many are girls?
Rationale: There are a total of 5 vans x 28 students = «5*28=140»140 students. If 60 are boys, then 140 - 60 = «140-60=80»80 of these students are girls. #### 80

B Used prompts

In the following box, we provide the prompt used to generate plausible rationales for the 25 few shot examples of ARC-Challenge, using GPT-4.

GPT-4 prompt for ARC rationale generation

You are expert grade-school science teacher. Given the following question, provide justification for the answer.
Question: `question`. Answer Choices: `options`. Answer: ... The Answer is `answer letter`.
You need to add a two to three sentences rationale before "The answer is `answer letter`", justifying the correct answer.

Next, we provide the template used to prompt the weak LLMs to generate rejected answers to create the preference dataset for Section 3.3.

Prompt template for weak LLM generation

You are an obedient assistant. Your task is to reason about the following question. Write only the next step of the reasoning chain. Your answer should include exactly one following reasoning step and has to be exactly one sentence long! The answer should start with "Next step: ". Here are two examples:

```
Question: {prompt_example_1}
Next step: {first_step_example_1}
Next step: {second_step_example_1}
Next step: {third_step_example_1}
Next step: {fourth_step_example_1}
Next step: {final_answer_example_1}

Question: {prompt_example_2}
Next step: {first_step_example_2}
Next step: {second_step_example_2}
Next step: {third_step_example_2}
Next step: {fourth_step_example_2}
Next step: {fifth_step_example_2}
Next step: {sixth_step_example_2}
Next step: {final_answer_example_2}

Question: {prompt}
Next step: {first_step_ground_truth}
...
```

C Training details

Number of epochs used for SFT: An important hyperparameter when doing supervised fine-tuning on small datasets is the number of times each example is processed. We optimized this hyperparameter independently using the smaller Falcon-7B (Almazrouei et al., 2023) as a base model, with the GSM8K accuracy as a metric, trying values $\{1, 2, 3, 4, 5\}$. We ended up using 3 epochs for all subsequent SFT experiments (using Falcon2-11B as a base model).

SFT learning rate: Similarly, using the GSM8K accuracy as a metric, and with a random search of a handful of learning rates in the range $[10^{-8}, 10^{-4}]$, we ended up using the learning rate of 1.4×10^{-5}.

prompt	chosen	rejected(1)	rejected(2)
Natalia sold clips to 48 of her friends in April, and then she sold half as many clips in May. How many clips did Natalia sell altogether in April and May?	Natalia sold 48/2 = 24 clips in May.	Natalia sold 32/4 = 19 clips in May.	Natalia sold 48 clips in April, and half as many clips in May, which is 24 clips
Natalia sold clips to 48 of her friends in April, and then she sold half as many clips in May. How many clips did Natalia sell altogether in April and May? Natalia sold 48/2 = 24 clips in May.	Natalia sold 48+24 = 72 clips altogether in April and May.	Natalia sold 25+98 = 12 clips altogether in April and May.	Natalia sold 24 clips in April.
Natalia sold clips to 48 of her friends in April, and then she sold half as many clips in May. How many clips did Natalia sell altogether in April and May? Natalia sold 48/2 = 24 clips in May. Natalia sold 48+24 = 72 clips altogether in April and May.	The solution to the problem is 72.	The solution to the problem is 13.	Natalia sold 24 clips in April, so she sold 24 clips in May.

Table 5: Example of preference dataset obtained from the **question** "Natalia sold clips to 48 of her friends in April, and then she sold half as many clips in May. How many clips did Natalia sell altogether in April and May?", and the corresponding **rationale** "Natalia sold $48/2 = 24$ clips in May. Natalia sold $48 + 24 = 72$ clips altogether in April and May. The solution to the problem is 72." This example is obtained from the GSM8K dataset (Cobbe et al., 2021). The **chosen** column represents steps from ground-truth rationale, **rejected (1)** are examples obtained by digit corruption, and **rejected (2)** are examples obtained by prompting the Llama-2-7B-chat model (Touvron et al., 2023)

SFT batch size: It is commonly agreed upon that larger batch sizes are more desirable when fine-tuning language models. We used a batch size of 16 as it was the largest that did not lead to memory issues on the GPUs we used.

LoRA parameters: Given the computational cost of fine-tuning LLMs, we chose not to tune the hyperparameters of LoRA (Hu et al., 2021), and resorted to using the popular values of $\alpha = 16$ and $rank = 64$.

Optimizer: For DPO, we used a linear schedule for the learning rate, and first jointly optimized the maximal learning rate and number of warm-up steps for the linear schedule, using RMSProp (Tieleman, 2012) as an optimizer. Optimizing this couple of hyperparameters was done on Falcon-7B using a preference dataset with digit corrupted rejected answers only, with the GSM8K accuracy as a metric. After settling on 10 warm-up steps, we tuned the maximal learning rate and the optimizer (choosing between RMSProp and AdamW (Loshchilov and Hutter, 2019)) on the same (model, task, metric) triplet. We ended up using the AdamW optimizer with a maximal learning rate of 8×10^{-6}.

Further DPO hyperparameters: We further optimized the learning rate, as well as the number of training epochs and the value of the β hyperparameter on the preference dataset $\mathcal{D}^{\text{pref}}_{\text{train}}$ constructed from GSM8K, and using digit-corrupted Llama-7B (Touvron et al., 2023), as explained in Section 3.3, to generate wrong answers. Using the resulting model's performance on the GSM8K task, with the evaluation protocol described in Section 3, we report the results of our hyperparameter sweep in Figure 3. This led to a universal choice of $\beta = 0.2$, learning rate of 8×10^{-6}, and number of epochs equal to 1 for all DPO experiments with Falcon2-11B.

Hyperparameters for DPO variants: Similar to DPO, the KTO (Ethayarajh et al., 2024) loss requires the specification of a hyperparameter β that controls how far the fine-tuned model drifts from the SFT model. IPO (Azar et al., 2023) needs a regularization parameter τ, for which the inverse τ^{-1} is usually denoted by β as well. For both methods, the value of β is critical and needs to be carefully tuned. We report in Figure 4 the results of our hyperparameter search. We consider the preference dataset $\mathcal{D}^{\text{pref}}_{\text{train}}$ constructed from GSM8K, and using digit corruption, as explained in Section 3.2, to generate wrong answers.
For ORPO (Hong et al., 2024), an important parameter is the weighing hyperparameter λ in **??**, that specifies the relative importance of the negative log-likelihood of the chosen answer with respect to the odds ratio part of the loss. We tried the values in the set $\{0.001, 0.005, 0.01, 0.1, 0.2, 0.3\}$ along with learning rates from the set $\{10^{-8}, 8 \times 10^{-8}, 8 \times 10^{-6}\}$, and found that $\beta = 0.001$ and 10^{-8} as a learning rate lead to the best results,

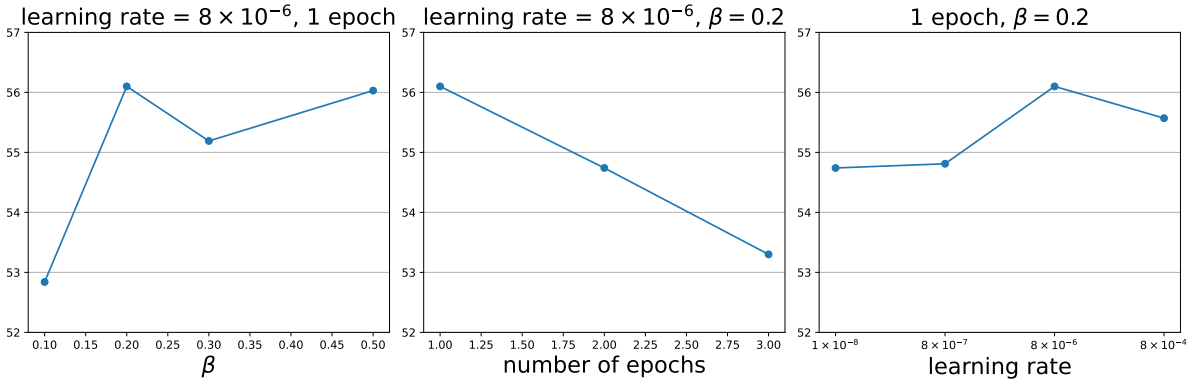

Figure 3: DPO hyperparameter search. The y axis corresponds to the accuracy on the test set of GSM8K.

which is what we report in Table 3.

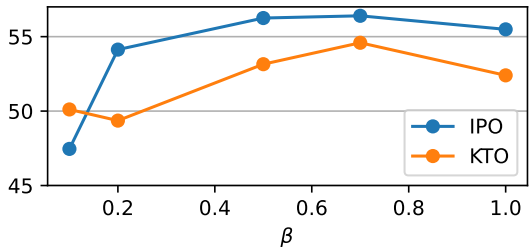

Figure 4: DPO variants hyperparameter search. The y axis corresponds to the accuracy on the test set of GSM8K. The learning rate 8×10^{-6} and number of epochs (1) used are the same as DPO.

D Additional Results

In Table 6, we study how the number of times each example from the GSM8K training dataset is visited during training affects the downstream performance on the related but different AQuA evaluation task. The table shows that reducing the training time could help dampen the overfitting issues of SFT.

	1 epoch	3 epochs
AQuA accuracy	33.46	30.71

Table 6: Accuracy (in percentage) on the AQuA test dataset of Falcon2-11B fine-tuned on $\mathcal{D}^{\text{SFT}}_{\text{train}}$ obtained from the GSM8K train dataset, as explained in Section 3.1. Comparison of the effect of number of epochs.

When using a weak LLM to generate rejected answers, it is not unlikely that the LLM outputs valid reasoning steps, in which case, including the resulting triplet in the preference dataset might hurt generalization of the resulting model. We therefore consider to corrupt the digits of the generated sequences similar to the digit corruption scheme alone. In Figure 5, we study the effect of post-generation digit corruption, and find that digit corruption is essential for downstream tasks. We also compare using the chat version of Llama-7B with the prompt template of Appendix B to using its base version, and find that using the base version yields to better performances.

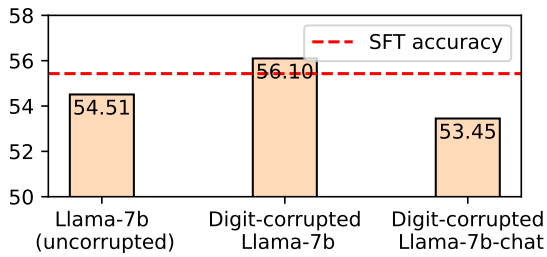

Figure 5: DPO with weak LLM generation for rejected answers. Comparison of different versions of Llama-7B. The y axis corresponds to the accuracy on the test set of GSM8K. The learning rate use is 8×10^{-6} and number of epochs is 1.

D.1 Ablations

We hypothesized that fine-tuning a language model to predict the next reasoning step only should help improve performances on reasoning benchmarks. Our results in the main paper confirm this hypothesis. However, it is natural to wonder whether using multiple reasoning steps to predict could be beneficial. More specifically, for SFT, we compare our approach (which requires fine-tuning on $(xz^{1:k-1}, z^k)$ pairs) to fine-tuning on $(xz^{1:k-1}, z^{k:n})$ pairs. Similarly, for DPO, we compare our approach (that requires fine-tuning on $(xz^{1:k-1}, z^k, \tilde{z}^k)$ triplets)

to fine-tuning on $(xz^{1:k-1}, z^{k:n}, \tilde{z}^k z^{k+1:n})$ triplets. Using Falcon2-11B as a base model, and GSM8K as a data source and for evaluation, we found that with this change, the performance drops from 55.43 to 54.81 for SFT, and from 58.30 to 57.01 for DPO with digit corruption.

We also tested replacing the inputs $xz^{1:k-1}$ with $uxz^{1:k-1}$, where u is a sequence corresponding to 3-shot examples, and found that while the SFT performance slightly increases to 55.95, the DPO performance significantly drops to 50.11.