Smaug: Fixing Failure Modes of Preference Optimisation with DPO-Positive

Arka Pal,* Deep Karkhanis, Samuel Dooley, Manley Roberts, Siddartha Naidu, Colin White

Abacus.AI

Abstract

Direct Preference Optimisation (DPO) is effective at significantly improving the performance of large language models (LLMs) on downstream tasks such as reasoning, summarisation, and alignment. Using pairs of preferred and dispreferred data, DPO models the *relative* probability of picking one response over another. In this work, first we show theoretically that the standard DPO loss can lead to a *reduction* of the model's likelihood of the preferred examples, as long as the relative probability between the preferred and dispreferred classes increases. We then show empirically that this phenomenon occurs when fine-tuning LLMs on common datasets, especially datasets in which the edit distance between pairs of completions is low. Using these insights, we design DPO-Positive (DPOP), a new loss function and training procedure which avoids this failure mode. Surprisingly, we find that DPOP outperforms DPO and other fine-tuning procedures across a wide variety of datasets and downstream tasks, including datasets with high edit distances between completions. Furthermore, we find that the DPOP-tuned model outperforms the DPO-tuned model (all else equal) on benchmarks independent of the fine-tuning data, such as MT-Bench. Finally, using DPOP, we create and open-source Smaug-34B and Smaug-72B, with the latter becoming the first open-source LLM to surpass an average accuracy of 80% on the HuggingFace Open LLM Leaderboard.

1 Introduction

Aligning large language models (LLMs) with preference data is important for their fluency and applicability to many tasks, with the natural language processing literature using many techniques to incorporate either human or ground-truth feedback [Christiano et al., 2017, Stiennon et al., 2020, Ouyang et al., 2022]. Typically in LLM alignment, we first collect large amounts of preference data, consisting of a context and two potential completions; one of these is labelled as the preferred completion, and the other as the dispreferred. We use this data to learn a general policy for generating completions in a given context. Direct Preference Optimisation (DPO) [Rafailov et al., 2023] is a popular method for learning from preference data, and it has shown to be effective at improving the performance of pretrained LLMs on downstream tasks such as reasoning, summarisation, and alignment [Wang et al., 2023, Tunstall et al., 2023]. The theoretical motivation for DPO is based on a preference-ranking model with an implicit reward function that models the *relative* probability of picking the preferred completion over the dispreferred.

---

*Correspondence to: `arka.pal@gmail.com`

Paired Preference Data

What is (3 + 5) / 2?

…3+5=**7**… < …3+5=**8**…

DPO (Rafailov et al. 2023)	**DPOP (ours)**
incentivise: log-prob on preferred > log-prob on dispreferred	incentivise: (i) log-prob on preferred > log-prob on dispreferred **and** (ii) log-prob on preferred $\geq$ ref log-prob on preferred

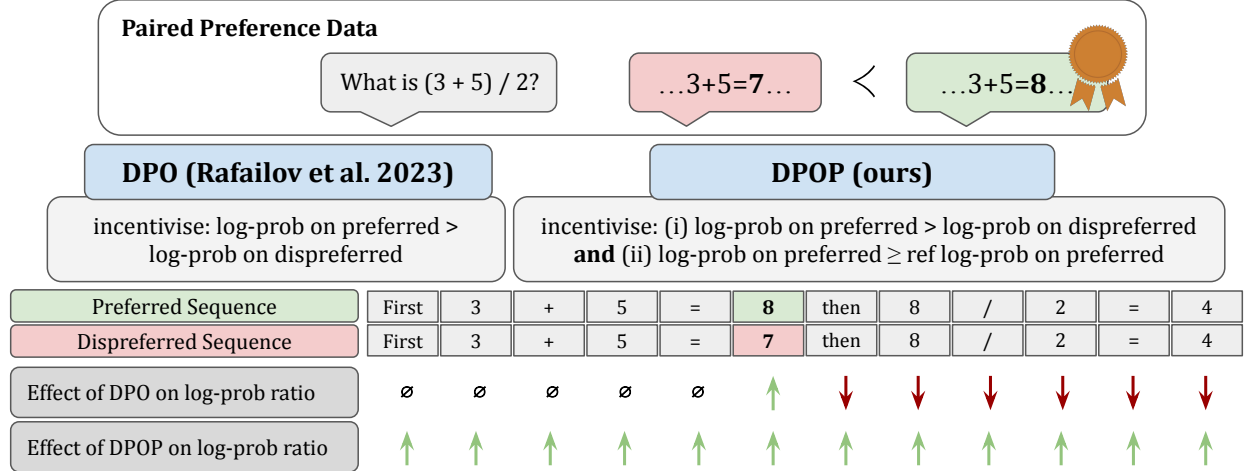| Preferred Sequence	First	3	+	5	=	**8**	then	8	/	2	=	4
Dispreferred Sequence	First	3	+	5	=	**7**	then	8	/	2	=	4
Effect of DPO on log-prob ratio	∅	∅	∅	∅	∅	↑	↓	↓	↓	↓	↓	↓
Effect of DPOP on log-prob ratio	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑

Figure 1: **DPOP avoids a failure mode of DPO.** When preference pairs differ on only a few tokens, DPO receives no loss incentive for the early tokens, and a loss incentive that in some cases can lead to degradation of the log-probs of later tokens (Section 3). We introduce DPOP, which adds a new term to the loss which leads every token to be incentivised toward the preferred completion (Section 4).

In this work, first we show theoretically that the standard DPO loss can lead to a *reduction* of the model's likelihood of the preferred completions, as long as the relative probability between the preferred and dispreferred classes increases. Our theoretical analysis suggests that the problem occurs most frequently in preference datasets with small edit distances between each pair of completions. Specifically, we show that in the case of preferred and dispreferred examples that differ by few tokens, DPO increases the probability of the token(s) that differ, yet decreases the probability of subsequent tokens (see Figure 1 for an overview). We also present an empirical token-level analysis that matches our theoretical findings on common datasets.

Using these insights, we design a new loss function: DPO-Positive (DPOP), which adds a new term to the loss function that penalises reducing the probability of the positive completions. We also create new preference datasets based on ARC [Clark et al., 2018], HellaSwag [Zellers et al., 2019], and MetaMath [Yu et al., 2023] and use them along with DPOP to create new models.

We introduce the Smaug class of models which use DPOP to achieve state-of-the-art open-source performance. We fine-tune 7B, 34B, and 72B models on our new datasets and show that DPOP far outperforms DPO. We evaluate our resulting models on multiple benchmarks including the HuggingFace Open LLM Leaderboard [Beeching et al., 2023, Gao et al., 2021], which aggregates six popular benchmarks such as MMLU [Hendrycks et al., 2021] and GSM8K [Cobbe et al., 2021]. On the HuggingFace Open LLM Leaderboard, Smaug-72B achieves an average accuracy of 80.48%, becoming the first open-source LLM to surpass an average accuracy of 80% and improving by nearly 2% over the second-best open-source model.

In order to address potential concerns about pervasive contamination on the HuggingFace Open LLM Leaderboard, we use an open-source contamination checker, finding that our model scores similarly to popular existing models. Then, we show that DPOP outperforms DPO in an apples-to-apples comparison on an LLM-judged benchmark that is independent of the fine-tuning data: MT-Bench [Zheng et al., 2023], a challenging benchmark representing eight different categories. We release our code, models, datasets, and documentation at `https://github.com/abacusai/smaug`.

Our contributions. We describe our main contributions below.

- We theoretically and empirically show a surprising failure mode of DPO: running DPO on preference datasets with small edit distances between completions can result in a catastrophic decrease in accuracy.
- We introduce DPO-Positive (DPOP) which we theoretically and empirically show ameliorates the performance degradation. In particular, DPOP often outperforms DPO, even on preference datasets with high edit distances between completions.
- We create new preference-based versions of ARC, HellaSwag, and MetaMath.
- Using DPOP and our new datasets, we create the Smaug series of models, with Smaug-72B becoming the first open-source model to achieve an average accuracy of 80% on the HuggingFace Open LLM Leaderboard. We open-source our trained models, datasets, and code.

2 Background and Related Work

Large language models (LLMs) have shown impressive zero-shot and few-shot performance [Radford et al., 2019, Brown et al., 2020, Bubeck et al., 2023]. Recently, researchers have fine-tuned pretrained LLMs on downstream tasks by using human-written completions [Chung et al., 2022, Mishra et al., 2021] or by using datasets labelled with human-preferred completions relative to other completions [Ouyang et al., 2022, Bai et al., 2022, Ziegler et al., 2020]. These techniques have been used to improve performance on a variety of downstream tasks such as translation [Kreutzer et al., 2018] and summarisation [Stiennon et al., 2020], as well as to create general-purpose models such as Zephyr [Tunstall et al., 2023]. Two of the most popular techniques for learning from preference data are reinforcement learning from human feedback (RLHF) [Ouyang et al., 2022, Bai et al., 2022, Ziegler et al., 2020] and direct preference optimisation (DPO) [Rafailov et al., 2023]. We summarise these approaches below.

RLHF   Consider a dataset of pairwise-preference ranked data $\mathcal{D} = \{x^{(i)}, y_w^{(i)}, y_l^{(i)}\}_{i=1}^N$ where $x^{(i)}$ are prompts and $y_w^{(i)}$ and $y_l^{(i)}$ are respectively the preferred and dispreferred completions conditioned on that prompt. We have an initial LLM π_{ref} that parameterises a distribution $\pi_{\text{ref}}(y|x)$. Often, we initialise π_{ref} as an LLM that has undergone supervised fine-tuning (SFT) to improve performance on downstream task(s). RLHF begins by modelling the probability of preferring y_w to y_l using the Bradley-Terry model [Bradley and Terry, 1952] which posits the following probabilistic form:

$$p(y_w \succ y_l|x) = \sigma(r(x, y_w) - r(x, y_l))$$

where σ is the logistic function and $r(x, y)$ corresponds to some latent reward function that is assumed to exist for the completion y given the prompt x. Given $\mathcal{D}$, we can learn a parameterised estimate of r by minimising the negative log-likelihood of the dataset:

$$\mathcal{L}_R(r_\phi, \mathcal{D}) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[\log(\sigma(r_\phi(x, y_w) - r_\phi(x, y_l)) \right].$$

For RLHF, we use reinforcement learning to optimise based on this learned reward function r_ϕ (with a regularising KL-constraint to prevent model collapse), and obtain a new LLM distribution π_θ.

DPO   Rafailov et al. [2023] showed that it is possible to optimise the same KL-constrained reward function as in RLHF without having to learn an explicit reward function. Instead, the problem is cast as a maximum likelihood optimisation of the distribution π_θ directly, with the objective:

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x, y_w, y_l) \sim D} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)} \right) \right] \tag{1}$$

where β is a regularisation term corresponding to the strength of KL-regularisation in RLHF. In this case, the implicit reward parameterisation is $r(x, y) = \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}$, and Rafailov et al. [2023] further showed that all reward classes under the Plackett-Luce model [Plackett, 1975, Luce, 2005] (such as Bradley-Terry) are representable under this parameterisation. For an abbreviation, we define $\pi_{\text{ratio}}(y|x) = \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}$.

Alternatives to DPO to align models to preference data in the offline, differentiable setting have been proposed. We discuss the most relevant to our work below and further in Appendix A.

SLiC Zhao et al. [2022, 2023] provides an alternative formulation for learning from pairwise preference data, either directly or on preference pairs sampled from the SFT policy. Their loss function is:

$$\mathcal{L}_{\text{SLiC}}(\pi_\theta) = \mathbb{E}_{(x,y_w,y_l)\sim D,y_{\text{ref}}}\left[\max(0, \beta - \log \pi_\theta(y_w|x) + \log \pi_\theta(y_l|x)) - \lambda \log \pi_\theta(y_{\text{ref}}|x)\right]$$

where y_{ref} is a reference target sequence and λ is a scalar. The second term in the above is a cross-entropy loss and plays the role of regularisation towards the original model, but without needing an extra copy of the weights.

IPO Azar et al. [2023] aims to understand the theoretical underpinnings of RLHF and DPO. They identify that DPO may be prone to overfitting in situations where the preference probability of the preferred over the dispreferred examples is close to 1. They propose an alternative form of pairwise preference loss—'Identity-PO (IPO)'. IPO tries to prevent overfitting to the preference dataset by penalising exceeding the preference margin beyond this regularised value. Conversely, we identify that DPO can lead to underfitting as well—even complete performance degradation.

Subsequent work Very recently, subsequent work has verified our main findings from Section 3 and Section 4 on the existence of a failure mode of DPO and how to fix it [Pang et al., 2024, Feng et al., 2024, Rafailov et al., 2024]. Pang et al. find that when fine-tuning on chain-of-thought reasoning tasks, adding a term similar to our Equation (3) is crucial to achieve strong performance [Pang et al., 2024]. Other work builds off of our theoretical insights by applying additional mathematical analysis, verifying our claims in Section 3 that DPO can decrease the probability of preferred completions [Feng et al., 2024]. Finally, Rafailov et al. analysed the same DPO failure mode phenomenon, showing that the likelihood of the preferred response should decrease during DPO training, when starting from a model that has undergone SFT [Rafailov et al., 2024]. We give additional discussion on related work in Appendix A.

3 Failure Mode of DPO

In this section, we take a step back and examine the DPO loss in Equation (1), specifically with an eye towards how it can reduce the probability of the preferred completion. The loss is a function only of the difference in the log-ratios, which means that we can achieve a low loss value even if $\pi_{\text{ratio}}(y_w|x)$ is lowered below 1, as long as $\pi_{\text{ratio}}(y_l|x)$ is also lowered sufficiently. This implies that the log-likelihood of the preferred completions is reduced below the original log-likelihood from the reference model.

Why is this an issue? The original use-case of RLHF did not explicitly denote the preferred completions as being also *satisfactory* completions (rather than just the preferred completion out of the two choices y_w and y_l), and hence the DPO objective is a good modelling choice. However, since then, a large body of work has focused on distilling the knowledge of powerful models into smaller or weaker models, while also

showing that doing so with RLHF/DPO outperforms SFT [Taori et al., 2023, Tunstall et al., 2023, Xu et al., 2023, Chiang et al., 2023]. In this paradigm, it is often the case that in each pair of completions, the better of the two is indeed also a satisfactory completion. Furthermore, a new technique is to transform a standard labelled dataset into a pairwise preference dataset [Ivison et al., 2023, Tunstall et al., 2023], which also has the property that for each pair of completions, one is a satisfactory completion.

Edit Distance 1   While the above illustrates a hypothetical situation, now we provide a specific case in which DPO may cause a decrease in the probability of the better completion. Consider the case of trying to improve a model's math or reasoning abilities by comparing a completion of "2+2=4" to "2+2=5." This process creates a pair of preferred and dispreferred completions which have an edit (Hamming) distance of 1, i.e., all tokens in the completion are the same except for one. In the following, we will explore how the location of the differing token impacts the computation of the DPO loss. For sake of argument, we will examine what happens when the differing token is the first token, though the argument also follows if it appears elsewhere.

For preliminaries, consider two completions with an edit distance of 1 which differ at token m with $1 \leq m \leq K$, i.e., consider $y_w = (t_1, \ldots, t_K)$ and $y_l = (t_1, \ldots, t_{m-1}, t'_m, t_{m+1}, \ldots, t_K)$. Denote $y^{<r} = (t_1, \ldots, t_{r-1})$ and $y^{\geq r} = (t_r, \ldots, t_K)$. Assume that the vocabulary length of the LLM is L. Let $s_i^{\{x\}}$ represent the probability of the i-th token in the model's vocabulary conditioned on the input x. While the LLM model parameters θ are numerous, we restrict our attention to the logits, θ_j with $j \in [L]$.

The gradient of Equation (1) with respect to θ is given by:

$$\nabla_\theta \mathcal{L}_{DPO}(\pi_\theta; \pi_{\text{ref}}) \propto -\nabla_\theta \left[\log \pi_\theta(y_w|x) - \log \pi_\theta(y_l|x) \right].$$

We note first that for any $m > 1$, all tokens in positions from 1 to $m-1$ have no effect on the gradient, as for all $1 \leq i \leq m-1$, $\pi_\theta(t_i|y_w^{<k}, x) = \pi_\theta(t_i|y_l^{<k}, x)$, causing these tokens' contribution to the gradient to cancel out. Therefore, without loss of generality, assume $m = 1$, i.e., y_w and y_l differ only at the first token. Without loss of generality, we also assume that t_k takes vocabulary position 1. Then we have the following for each $k > 1$ (derivation in Appendix B.1):

$$\nabla_{\theta_j} \left[\log \pi_\theta(y_w|x) - \log \pi_\theta(y_l|x) \right] = s_j^{\{y_l^{<k}, x\}} - s_j^{\{y_w^{<k}, x\}}. \tag{2}$$

As we typically run DPO after SFT, the model is likely to be reasonably well optimised, so we should have $s_j^{\{y_w^{<k}, x\}} \leq s_j^{\{y_l^{<k}, x\}}$ for $j \neq 1$ and $s_1^{\{y_w^{<k}, x\}} \geq s_1^{\{y_l^{<k}, x\}}$. Therefore, while this analysis only extends to gradients with respect to the logits, we see that the gradient decreases logits corresponding to the correct token and increases logits corresponding to the incorrect tokens. Surprisingly, this suggests that under DPO, all locations in the sequence after the differing token will have reduced probability of emitting the correct token when compared to π_{ref}. We give empirical evidence for this in Section 5 and Figure 4.

4   DPOP

Now, we introduce DPO-Positive (DPOP), our proposed method to address the failure mode described in the previous section. DPOP adds the penalty term $\max \left(0, \log \frac{\pi_{\text{ref}}(y_w|x)}{\pi_\theta(y_w|x)} \right)$ to the loss within the log-sigmoid to incentivise maintaining a high log-likelihood of the preferred completions. This penalty term is 0 when $\pi_{\text{ratio}}(y_w|x) \geq 1$ and increases as the ratio goes below 1.

The full DPOP loss function is thus:

$$\mathcal{L}_{\text{DPOP}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim D}\left[\log\sigma\left(\beta\left(\log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \log\frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right.\right.\right.$$
$$\left.\left.\left. - \lambda\cdot\max\left(0, \log\frac{\pi_{\text{ref}}(y_w|x)}{\pi_\theta(y_w|x)}\right)\right)\right)\right] \quad (3)$$

where $\lambda > 0$ is a hyperparameter that can be tuned. This form of loss retains the property that we are fitting parameters on the preference data under the Bradley-Terry model. The implicit reward parameterisation is

$$\beta\cdot\log\frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)} \text{ for } y = y_l, \qquad \beta\left[\log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y|x)} - \lambda\cdot\max\left(0, \log\frac{\pi_{\text{ref}}(y|x)}{\pi_\theta(y|x)}\right)\right] \text{ for } y = y_w.$$

By applying this optimisation pressure, the model can no longer minimise the loss by significantly reducing the log-likelihood of the dispreferred examples more than it reduces the log-likelihood of the preferred examples; it must also ensure that the log-likelihood of the preferred examples remains high relative to the log-likelihood under the reference model.

Now we show that Equation (3) mitigates the failure mode from the previous section. Recall from Section 3 that we focused on two completions, y_w and y_l, which differ by one token at location $m = 1$. We showed in Equation (2) that for standard DPO, the gradient of the k-th token in the completions with respect to the j-th logit is $s_j^{\{y_l^{<k},x\}} - s_j^{\{y_w^{<k},x\}}$. However, for DPOP, if $\pi_{\text{ratio}} < 1$, the gradients become (derivation in Appendix B.2):

$$\nabla_{\theta_j}\left[\log\pi_\theta(t_k|y_w^{<k},x) - \log\pi_\theta(t_k|y_l^{<k},x) + \lambda\cdot\log\pi_\theta(t_k|y_w^{<k},x)])\right]$$
$$= \begin{cases} \lambda\left(1 - s_j^{\{y_w^{<k},x\}}\right) + s_j^{\{y_l^{<k},x\}} - s_j^{\{y_w^{<k},x\}} & i = j \\ -(\lambda+1)s_j^{\{y_w^{<k},x\}} + s_j^{\{y_l^{<k},x\}} & i \neq j \end{cases}$$

where i is the vocabulary index of token t_k. Since $s_j^{\{y_w^{<k},x\}} \leq 1$, for the case $i = j$, the gradient is guaranteed to be positive for a large enough choice of λ. Similarly, for the case $i \neq j$, the gradient is guaranteed to be negative for a large enough λ (as long as $s_j^{\{y_w^{<k},x\}} > 0$). This therefore fixes the issue we identified in DPO in Section 3.

Connection to Contrastive Loss While the main motivation for DPOP is to avoid the failure mode described in Section 3, we also note its connection to *contrastive loss*. Contrastive learning is a popular technique in areas such as computer vision for datasets of similar and dissimilar pairs [Oord et al., 2018, Chen et al., 2020, He et al., 2020], and the loss function often uses a margin factor. Equation (3) can be viewed as similar to contrastive loss with margin $m = \log\frac{1}{\pi_{\text{ref}}(y_w|x)}$. We give further details in Appendix C.

5 DPOP Datasets & Experiments

In this section, we empirically validate that the failure mode *does* arise in practice and that DPOP is able to mitigate the failure. We also show that even when the edit distance is large and DPO does not show degradation in performance, DPOP can still outperform DPO on downstream task evaluation.
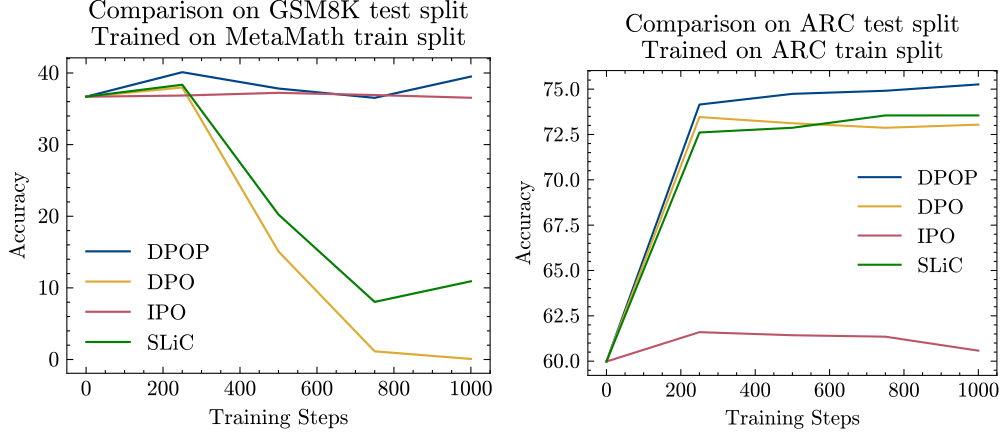
Figure 2: **Preference optimization comparisons on low (MetaMath) and high (ARC) edit distance datasets**. DPOP outperforms DPO, IPO, and SLiC on both MetaMath (left) and ARC (right), whose normalized edit distances are 6.5% and 90%, respectively. Evaluation is performed on the test set of the datasets using the LLM Evaluation Harness.

5.1 Dataset Creation

For our empirical analysis, we focus on the downstream tasks of **GSM8K**, **ARC**, and **HellaSwag**, and we introduce and release associated paired preference-ranked datasets.

GSM8K [Cobbe et al., 2021], a dataset of diverse grade school math word problems, has been adopted as a measure of the math and reasoning skills of LLMs [Chowdhery et al., 2023, Touvron et al., 2023b,a, Beeching et al., 2023, Gao et al., 2021]. We create a paired preference-ranked version of MetaMath [Yu et al., 2023], an extended version of the GSM8K training data [An et al., 2023, Yu et al., 2023]. The correct completions in the MetaMath dataset consist of a series of steps which lead to the final answer. To create a dispreferred version, we randomly corrupt one of the results of an intermediate calculation. This dataset has a low (normalised) edit distance of 6.5%.

ARC [Clark et al., 2018] is a dataset that tests the level of understanding of science at grade-school level. We focus specifically on ARC-Challenge, the more difficult of the two subsections of ARC, which has been widely adopted as a measure of LLM reasoning and world understanding [Chowdhery et al., 2023, Touvron et al., 2023b,a, Beeching et al., 2023, Cobbe et al., 2021]. The ARC-Challenge dataset consists of four choices of responses to each question, one of which is correct. To create a paired preference-ranked dataset, for each correct response in the training split, we create three pairs using each incorrect response. Due to the differences in the responses, this dataset has a high normalised edit distance of 90%.

HellaSwag [Zellers et al., 2019] is a dataset containing commonsense inference questions known to be hard for LLMs. Similar to ARC, each question has one correct completion and three incorrect completions, and so we create a paired preference-ranked dataset by creating three pairs for each correct response in the training split. See Appendix D for further details and documentation about our new datasets.

5.2 Experiments

In this section, we compare training DPO, IPO, SLiC and DPOP on the train splits of the datasets mentioned above and evaluate them on the corresponding tasks. We apply each preference-training method to the base
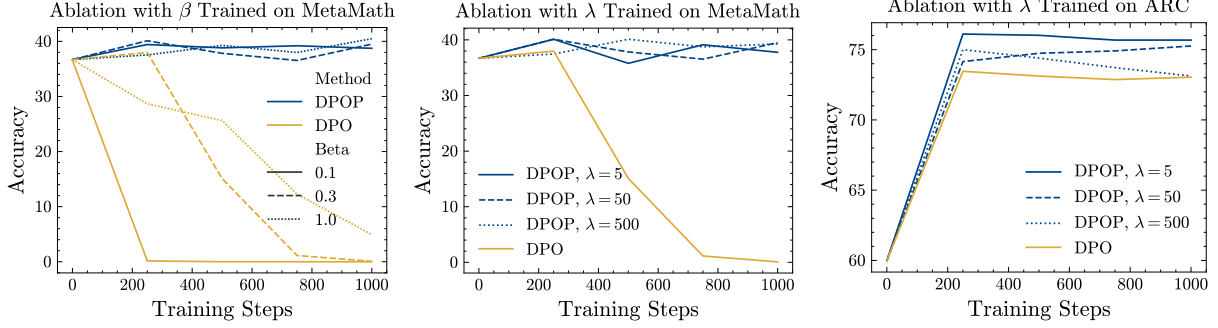
Figure 3: **Ablation studies:** Evaluation of DPO vs. DPOP for different values of β, on the MetaMath dataset (left), as well as for different values of λ, on the MetaMath (middle) and ARC (right) datasets.

model of Mistral7B [Jiang et al., 2023]. We evaluate on the test sets of GSM8K and ARC with the LLM Evaluation Harness [Gao et al., 2021]. Unless specified otherwise, we use values of $\beta = 0.3$ and $\lambda = 50$.

Loss function comparison First, we compare DPO, IPO, SLiC and DPOP when training on both MetaMath and ARC; see Figure 2. We find that when training on MetaMath, both DPO and SLiC catastrophically fail, while IPO does not improve performance. DPOP is the only model to improve performance over the base model. When training on ARC, which has a higher edit distance as described in the previous section, both DPO, SLiC and DPOP are able to improve on the base model significantly; however, DPOP performs best.

Ablation studies for β and λ One potential hypothesis for how degradation of DPO on MetaMath could be prevented is by modifying the strength of the regularisation parameter, β. We test $\beta \in \{0.1, 0.3, 1.0\}$, and although a larger β does induce a slower decrease, the performance with DPO still plummets, while DPOP shows strong and consistent performance with different values of β (see Figure 3). Furthermore, we conduct an ablation study over the value of λ (a hyperparameter unique to DPOP) to determine the sensitivity of performance to tuning this value precisely. We test $\lambda \in \{5, 50, 500\}$ and find that performance on MetaMath and ARC is relatively unaffected by the choice of λ. See Figure 3.

Token-level analysis Recall that in Section 3, we gave theoretical motivations for why DPO is likely to perform poorly on low-edit distance datasets. We now analyse the log-probabilities of the trained models at the token level on the MetaMath dataset over 900 samples to empirically support our arguments. Let us denote the index of the first token that is different between the preferred and dispreferred completion by m.

We suggested that $\pi_\theta(y^{\geq r} \mid x, y^{<r})$ for $r > m$ will have 'wrong-way' gradient updates and therefore decrease. We find this is indeed the case—the average log-prob after training of tokens after m is -0.37 for the reference model and -0.26 for DPOP, but -1.82 for DPO on the preferred completions (see (Figure 4) (left)). Perhaps most instructively, for both the reference model and DPOP, in Figure 4 (right), we see that tokens after the edit indices show higher log-likelihood than those before the edit indices—this is indicative of well-behaved language modelling, with lower perplexity as more tokens are added to the context. By contrast, DPO shows the opposite pattern—with log-likelihood actually reducing after the edit indices. This is indicative of a deeper breakdown in language modelling, which we believe is facilitated by the wrong-way gradient we outlined in Section 3. Finally, we are also able to substantiate our assumption from Section 3 that $s_1^{\{y_w^{<k}, x\}} \geq s_1^{\{y_l^{<k}, x\}}$; we find from our analysis that for the baseline model, the tokens after the edit have
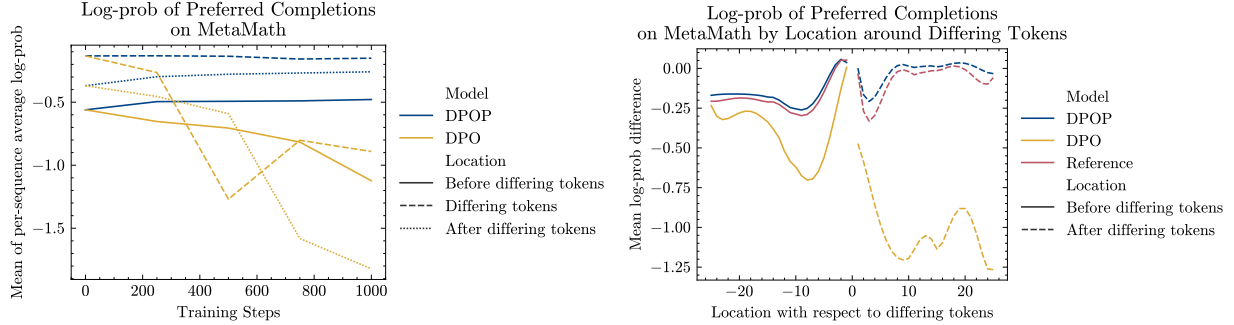
Figure 4: **DPO fails to train on low edit-distance pairs, yet DPOP performs well.** Left: average log-probs for 900 randomly-sampled preferred train set completions on MetaMath across training steps. For DPO, the log-probs decrease throughout training. Right: average log-prob difference for preferred completions on MetaMath by location around differing tokens, after 1000 training steps. Log-prob 'difference' signifies that each model's plot has been adjusted to have 0 log-prob at location -1; all other log-probs are shown relative to this value. For DPO, there is a significant decrease after the differing tokens, while DPOP avoids this issue. The log-probs of the LLM prior to application of DPOP or DPO is also included for reference.

an average log-likelihood of -0.37 on the preferred completion, but this drops to -0.86 on the dispreferred completion.

In Appendix E, we present additional results on ARC comparing the averaged log-probs of DPO and DPOP on the preferred completion during training; see Appendix Figure 6. DPOP once again demonstrates higher log-probs than DPO.

6   Smaug

In this section, we introduce the Smaug series of models. We train models for 7B, 34B and 72B parameter sizes using DPOP. We use the 7B class for a direct comparison of DPOP vs. DPO, including testing the generalisability of the improvement in model performance on benchmarks very different from the training data, such as MT-Bench [Zheng et al., 2023]. Due to the computational resource requirements involved in training the larger model sizes, we only perform DPOP on 34B and 72B and compare to other models on the HuggingFace Open LLM Leaderboard. Also due to computational expense, we do not perform any further hyperparameter tuning.

Datasets   In this section, unless otherwise noted, we train on a mix of six pair preference datasets: our MetaMath DPO, ARC DPO, and HellaSwag DPO datasets described in Section 5, the ORCA DPO dataset [Intel, 2024], the Truthy DPO dataset [Durbin, 2024b], and the UltraFeedback Binarized dataset [AllenAI, 2024]. We run these experiments with 8 H100 GPUs (each with 80GB), using the `transformers` repository [Wolf et al., 2020] for general model loading and infrastructure, and the TRL repository [von Werra et al., 2020] for running DPOP. We set $\beta = 0.3$, $\lambda = 50$, a learning rate of 5×10^{-5}, and the AdamW optimizer [Loshchilov and Hutter, 2017], and we run 1000 steps for all DPO and DPOP routines.

6.1 DPOP vs. DPO Ablation on 7B

First we apply the above recipe on Llama-2-Chat [Touvron et al., 2023b]. Since this model has already undergone instruction fine-tuning, we perform DPO and DPOP directly using the datasets described in the previous section. We evaluate the Llama-2 finetunes on MT-Bench [Zheng et al., 2023] This benchmark tests across multiple categories of LLM performance (for example: writing, roleplay, coding and math).

MT-Bench We evaluate MT-Bench on Llama-2-7B-Chat finetuned with DPO and DPOP. We run MT-Bench with the Llama-2 conversation template. We perform 10 trials due to the inherent stochasticity of the benchmark, computing mean and standard deviation. Across the 10 trials, DPO achieves a first turn score of 7.032 ± 0.043 whilst DPOP scores 7.292 ± 0.037, a significant improvement.

MT-Bench tests across eight diverse categories and is significantly different from our fine-tuning data. Furthermore, MT-Bench has been shown to have performance correlated to human preferences of LLM rankings [Zheng et al., 2023, Rafailov et al., 2023, Li et al., 2024]. The outperformance of DPOP vs. DPO in this controlled like-for-like setting is therefore an indication of the improvement gained by utilising DPOP over DPO on new tasks not in the training set, and also an indication of general model quality.

6.2 Smaug-34B and Smaug-72B

Smaug-34B is a modified version of the base model Bagel-34B-v0.2 [Durbin, 2024a], which itself is a SFT version of Yi-34B-200k [01.AI, 2024]. We first take Bagel-34B-v0.2 and perform a SFT fine-tune using a combination of three datasets: MetaMath [Yu et al., 2023], ORCA-Chat [Es, 2024], and the ShareGPT dataset [Z., 2024]. We then apply DPOP using the methodology and datasets described above.

For 72B, we start from MoMo-72b-lora-1.8.7-DPO [Moreh, 2024], which itself is a fine-tune of Qwen-72B [Bai et al., 2023]. MoMo-72b-lora-1.8.7-DPO has already undergone SFT, so we simply run the DPOP routines as in Smaug-34B. The total training time is 144 hours.

HuggingFace Open LLM Leaderboard We evaluate using the HuggingFace Open LLM Leaderboard [Beeching et al., 2023, Gao et al., 2021], a widely-used benchmark suite that aggregates six popular benchmarks: ARC [Clark et al., 2018], GSM8K [Cobbe et al., 2021], HellaSwag [Zellers et al., 2019], MMLU [Hendrycks et al., 2021], TruthfulQA [Lin et al., 2022], and WinoGrande [Sakaguchi et al., 2020]. We evaluate directly in HuggingFace, which uses the LLM Evaluation Harness [Gao et al., 2021]. It performs few-shot prompting on the test sets of these tasks. We compare Smaug-72B to the evaluation scores of the top five open-weight LLMs according to the HuggingFace Open LLM Leaderboard [Beeching et al., 2023, Gao et al., 2021] as of March 2024; see Table 1. Smaug-72B achieves an average accuracy of 80.48%, becoming the first open-source LLM to surpass an average accuracy of 80% and improving by nearly 2% over the second-best open-source model. Smaug-34B also achieves best-in-its-class performance compared to other models of similar or smaller size (see Appendix E).

MT-Bench Next, we evaluate again using MT-Bench [Zheng et al., 2023]. We run MT-Bench with the Llama-2 conversation template [Touvron et al., 2023b]. See Appendix Table 2 for a comparison with state-of-the-art LLMs according to Arena Elo as of March 2024. Smaug-72B achieves the top MMLU score and third-best MT-bench score out of the open-source models. In Appendix E, we give examples of Smaug-72B completions to MT-Bench questions.

Table 1: Evaluation of the top open-weight models on the HuggingFace Open LLM Leaderboard as of March 2024. See Table 4 for an extended comparison.

Model	Size	Avg.	ARC	HellaSwag	MMLU	TruthfulQA	Winogrande	GSM8K
Smaug-72B (Ours)	72B+	**80.48**	**76.02**	89.27	**77.15**	76.67	85.08	**78.70**
MoMo-72B-lora-1.8.7-DPO	72B+	78.55	70.82	85.96	77.13	74.71	84.06	78.62
TomGrc_FusionNet_34Bx2_MoE_v0.1_DPO_f16	72B+	77.91	74.06	86.74	76.65	72.24	83.35	74.45
TomGrc_FusionNet_34Bx2_MoE_v0.1_full_linear_DPO	72B+	77.52	74.06	86.67	76.69	71.32	83.43	72.93
Truthful_DPO_TomGrc_FusionNet_7Bx2_MoE_13B	72B+	77.44	74.91	**89.30**	64.67	**78.02**	**88.24**	69.52

6.3 Contamination check

Since LLMs train on wide swaths of the internet, data contamination, i.e., evaluating on examples that are included in the training data, is a growing concern. Data contamination remains notoriously challenging to measure and mitigate, even with partial attempts like controlled experimentation of training data, canary strings, or embedding similarities [Roberts et al., 2024, Jain et al., 2024, bench authors, 2023].

While there is no perfect tool to check for data contamination, we use an open-source contamination checker [Shi, 2023] to compare the contamination of our model to other open-source models on training datasets of the Huggingface Open LLM Leaderboard. On ARC, TruthfulQA, and GSM8K, we find that Smaug-72B achieves scores that are similar to MoMo-72B-lora-1.8.7-DPO (the starting point of Smaug-72B), as well as Llama-2-70B. See the full details in Appendix E.3.

7 Conclusions and Limitations

In this work, we presented new theoretical findings on a severe failure mode of DPO, in which fine-tuning causes the probability of the preferred examples to be reduced. We then presented an empirical token-level analysis that matches our theoretical findings on popular datasets. In order to mitigate this issue, we devised a new technique, DPOP, which we show overcomes the failure mode of DPO — and can outperform DPO even outside this failure mode. By fine tuning with DPOP on our new pairwise preference versions of ARC, HellaSwag, and MetaMath, we create a new LLM that achieves state-of-the-art performance. In particular, we present the first open-weights model to surpass an average accuracy of 80% on the HuggingFace Open LLM Leaderboard, and we also show that DPOP significantly outperforms DPO in an apples-to-apples comparison on MT-Bench, which is independent of the fine-tuning data.

In the future, creating pairwise preference-based versions of other datasets, and running DPOP with these datasets, could push the abilities of open-source LLMs closer to the performance of proprietary models such as GPT-4 [OpenAI, 2023], especially when tuned for specific downstream tasks. Furthermore, using DPOP on additional mathematical datasets is an exciting area for future work, as it has the potential to further advance LLMs' abilites in mathematical reasoning.

Limitations While our work gives theoretical and empirical evidence on a failure mode of DPO and a proposed solution, it still has limitations. First, we were unable to run a full ablation study on our 72B model. Running multiple fine-tuning experiments on a 72B model is infeasible, as each one can take over five days to complete. Therefore, we assume that our ablations on smaller models still hold up at scale. Furthermore, while we expect DPOP to achieve strong performance on any preference dataset, especially those with small edit distance, we have only demonstrated its performance on six English-language datasets. We hope that future work can verify its effectiveness on more datasets, in particular on non-English datasets.

References

01.AI. Yi-34b-200k, 2024. URL https://huggingface.co/01-ai/Yi-34B-200K.

AllenAI. Ultrafeedback binarized clean, 2024. URL https://huggingface.co/datasets/allenai/ultrafeedback_binarized_cleaned.

Shengnan An, Zexiong Ma, Zeqi Lin, Nanning Zheng, Jian-Guang Lou, and Weizhu Chen. Learning from mistakes makes llm better reasoner. *arXiv preprint arXiv:2310.20689*, 2023.

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

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022.

Edward Beeching, Clémentine Fourrier, Nathan Habib, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. Open llm leaderboard. https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard, 2023.

BIG bench authors. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models, 2023.

R. A. Bradley and M. E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952. doi: 10.2307/2334029.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

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

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*, 2022.

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

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

Jon Durbin. Bagel-34b-v0.2, 2024a. URL https://huggingface.co/jondurbin/bagel-34b-v0.2.

Jon Durbin. Truthy dpo, 2024b. URL https://huggingface.co/datasets/jondurbin/truthy-dpo-v0.1.

Shahul Es. Orca-chat, 2024. URL https://huggingface.co/datasets/shahules786/orca-chat.

Kawin Ethayarajh, Winnie Xu, Dan Jurafsky, and Douwe Kiela. Human-centered loss functions (halos). Technical report, Contextual AI, 2023.

Duanyu Feng, Bowen Qin, Chen Huang, Zheng Zhang, and Wenqiang Lei. Towards analyzing and understanding the limitations of dpo: A theoretical perspective. *arXiv preprint arXiv:2404.04626*, 2024.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, September 2021.

R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742, 2006. doi: 10.1109/CVPR.2006.100.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, et al. Measuring coding challenge competence with apps. *arXiv preprint arXiv:2105.09938*, 2021.

Intel. Orca dpo pairs, 2024. URL https://huggingface.co/datasets/Intel/orca_dpo_pairs.

Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A Smith, Iz Beltagy, et al. Camels in a changing climate: Enhancing lm adaptation with tulu 2. *arXiv preprint arXiv:2311.10702*, 2023.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

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. Mistral 7b, 2023.

Julia Kreutzer, Joshua Uyheng, and Stefan Riezler. Reliability and learnability of human bandit feedback for sequence-to-sequence reinforcement learning. *arXiv preprint arXiv:1805.10627*, 2018.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. From live data to high-quality benchmarks: The arena-hard pipeline, April 2024. URL https://lmsys.org/blog/2024-04-19-arena-hard/.

Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2022.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

R Duncan Luce. *Individual choice behavior: A theoretical analysis*. Courier Corporation, 2005.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. Cross-task generalization via natural language crowdsourcing instructions. *arXiv preprint arXiv:2104.08773*, 2021.

Moreh. Momo-72b-lora-1.8.7-dpo, 2024. URL https://huggingface.co/moreh/MoMo-72B-lora-1.8.7-DPO.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

OpenAI. Gpt-4 technical report. *Technical Report*, 2023.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. 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 Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022.

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

Robin L Plackett. The analysis of permutations. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 24(2):193–202, 1975.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

Rafael Rafailov, Joey Hejna, Ryan Park, and Chelsea Finn. From r to q* : Your language model is secretly a q-function. *arXiv preprint arXiv:2404.12358*, 2024.

Manley Roberts, Himanshu Thakur, Christine Herlihy, Colin White, and Samuel Dooley. To the cutoff... and beyond? a longitudinal perspective on llm data contamination. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34, 2020.

Nikunj Saunshi, Orestis Plevrakis, Sanjeev Arora, Mikhail Khodak, and Hrishikesh Khandeparkar. A theoretical analysis of contrastive unsupervised representation learning. In *International Conference on Machine Learning*, pages 5628–5637. PMLR, 2019.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823, 2015. doi: 10.1109/CVPR.2015.7298682.

Weijia Shi. Detect pretrain code contamination. https://github.com/swj0419/detect-pretrain-code-contamination, 2023.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html*, 3(6):7, 2023.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023a.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023b.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. Zephyr: Direct distillation of lm alignment, 2023.

Amos Tversky and Daniel Kahneman. Advances in prospect theory: Cumulative representation of uncertainty. *Journal of Risk and Uncertainty*, 5(4):297–323, 1992.

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

Feng Wang and Huaping Liu. Understanding the behaviour of contrastive loss. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2495–2504, 2021.

Peiyi Wang, Lei Li, Liang Chen, Feifan Song, Binghuai Lin, Yunbo Cao, Tianyu Liu, and Zhifang Sui. Making large language models better reasoners with alignment. *arXiv preprint arXiv:2309.02144*, 2023.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR, 2020.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*, 2023.

Haoran Xu, Amr Sharaf, Yunmo Chen, Weiting Tan, Lingfeng Shen, Benjamin Van Durme, Kenton Murray, and Young Jin Kim. Contrastive preference optimization: Pushing the boundaries of llm performance in machine translation. *arXiv preprint arXiv:2401.08417*, 2024.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models, 2023.

Z. Sharegpt_vicuna_unfiltered, 2024. URL https://huggingface.co/datasets/anon8231489123/ShareGPT_Vicuna_unfiltered.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1472. URL https://www.aclweb.org/anthology/P19-1472.

Yao Zhao, Misha Khalman, Rishabh Joshi, Shashi Narayan, Mohammad Saleh, and Peter J. Liu. Calibrating sequence likelihood improves conditional language generation, 2022.

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

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*, 2023.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences, 2020.

A   Related Work Continued

In this section, we continue our discussion of related work from Section 2.

AFT   Wang et al. [2023] seek to align LLMs to correctly 'score' (in terms of perplexity) their own generations. They do so by generating multiple chain-of-thought [Wei et al., 2023] responses to each prompt, which they categorise as preferred or dispreferred according to whether they answer the question correctly. Their proposed 'Alignment Fine-Tuning (AFT)' paradigm adds an alignment objective $\mathcal{L}_A^*$ to the standard fine tuning loss, defined as

$$\mathcal{L}_A^*(\pi_\theta) = \log\left[1 + \sum_{y_w \in \mathcal{G}_W} \sum_{y_l \in \mathcal{G}_L} e^{(\log \pi_\theta(y_l|x) - \log \pi_\theta(y_w|x))}\right]$$

where $\mathcal{G}_p$ is the set of preferred examples and $\mathcal{G}_n$ is the set of dispreferred examples. By minimising $\mathcal{L}_A^*$, the log-likelihoods of preferred examples are encouraged to be larger than the log-likelihoods of dispreferred examples, akin to DPO. However, Wang et al. [2023] takes an opposing motivation to us: they are particularly concerned with the issue of the log-likelihoods of dispreferred examples being pushed down too significantly

Our work differs from AFT in three key points. First, although Wang et al. [2023] discusses DPO in the appendix, they do not show how their approach would extend to a reformulation of its objective; they also focus their experiments solely on supervised fine-tuning. Next, we use a different constraint mechanism—theirs is a soft margin constraint on the log-probability distance of the dispreferred example from the preferred example, while ours is a soft penalty for deviating from a reference model. Finally, they are focused specifically on the case of self-generated LLM CoT responses and calibrating the LLM's perplexity of its own responses.

HALO   Ethayarajh et al. [2023] seek to understand alignment methods, including DPO, in the context of 'Human-Centred Loss Functions (HALOs)'. By drawing an equivalence between the alignment methods and the work of Tversky and Kahneman [1992] in prospect theory, they adapt the 'human value function' in that paper to the LLM setting:

$$L_{KTO}(\pi_\theta, \pi_{\text{ref}}) = \mathbb{E}_{x,y \sim D}\left[w(y)(1 - \hat{h}(x, y; \beta))\right]$$

where they define $g(x, y; \beta)$ as

$$\beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)} - \mathbb{E}_{x' \sim D}\left[\beta \mathrm{KL}(\pi_\theta||\pi_{\text{ref}})\right]$$

and

$$h(x, y; \beta) = \begin{cases} \sigma(g(x, y; \beta)) & \text{if } y \sim y_w|x \\ \sigma(-g(x, y; \beta)) & \text{if } y \sim y_l|x \end{cases}$$

$$w(y) = \begin{cases} \lambda_D & \text{if } y \sim y_w|x \\ \lambda_U & \text{if } y \sim y_l|x \end{cases}$$

The major difference of this approach with DPO is that it does not require paired preference data. The above loss function can be used for any dataset as long as the labels are individually marked as positive or negative.

CPO   Very recently, concurrent work [Xu et al., 2024] proposes adding a new term to the DPO loss function in order to allow DPO to become better at rejecting 'worse' completions that are good quality but not perfect. Specifically, they include the term

$$\mathbb{E}_{(x,y_w,y_l)\sim D}\left[\log \pi_\theta(y_w \mid x)\right].$$

While similar, their work uses a different loss function with different motivation, and furthermore they only considered machine translation models up to 13B parameters.

Subsequent work, extended discussion   As described in Section 2, very recently, subsequent work has verified our main findings from Section 3 and Section 4 on the existence of a failure mode of DPO and how to fix it [Pang et al., 2024, Feng et al., 2024, Rafailov et al., 2024]. We give additional discussion here.

Pang et al. consider fine-tuning on chain-of-thought reasoning tasks using pairwise preference data. They find that including an additional positive log-likelihood term is crucial for their reasoning tasks. Their final loss is similar to our Equation (3), but the additional term is outside the log sigmoid [Pang et al., 2024].

Feng et al. give additional mathematical analyses of the gradient vector fields of the DPO loss. They come to the conclusion that the DPO loss function decreases the probability of producing dispreferred completions at a faster rate than it increases the probability of producing preferred completions [Feng et al., 2024].

Finally, Rafailov et al. also analysed the "phenomena in which the likelihood of the chosen responses actually decrease over time [in DPO]" by examining the expected log ratio of a completion under a model being optimised to a reference model. By showing that this is equivalent to the KL divergence, which is necessarily non-negative, they find that "the likelihood of the chosen response should decrease in the process of DPO training" when starting from a model that has undergone SFT on the preferred completions [Rafailov et al., 2024].

B   Derivation of logit gradients

B.1   Derivation for DPO

Consider two completions of length K with edit (Hamming) distance of 1 which differ at token m with $1 \leq m \leq K$. Put $y_w = (t_1, \ldots, t_K)$ and $y_l = (t_1, \ldots, t_{m-1}, t'_m, t_{m+1}, \ldots, t_K)$. Put $y^{<r} = (t_1, \ldots, t_{r-1})$ and $y^{\geq r} = (t_r, \ldots, t_K)$. Note that the derivative of Equation (1) with respect to θ is proportional to:

$$\nabla_\theta \mathcal{L}_{DPO}(\pi_\theta; \pi_{\text{ref}}) \propto -\left[\nabla_\theta \log \pi_\theta(y_w|x) - \nabla_\theta \log \pi_\theta(y_l|x)\right]$$

By the chain rule, we have

$$\pi_\theta(y|x) = \prod_{k=1}^{K} \pi_\theta(t_k|y^{<k}, x)$$

and by using logarithm identities

$$\log \pi_\theta(y|x) = \sum_{k=1}^{K} \log \pi_\theta(t_k|y^{<k}, x).$$

When we substitute this into $\nabla_\theta \mathcal{L}_{DPO}$, we have:

$$\nabla_\theta \mathcal{L}_{DPO}(\pi_\theta; \pi_{\text{ref}}) \propto - \sum_{k=1}^{K} \nabla_\theta \left[\log \pi_\theta(t_k|y_w^{<k}, x) - \log \pi_\theta(t_k|y_l^{<k}, x) \right] \tag{4}$$

Assume that the vocabulary length of the LLM is L. While the LLM model parameters θ are numerous, let us restrict our attention to just the logits, which we denote as θ_j with $j \in [L]$ and are the input to the softmax. We further denote the softmax probabilities by $s^{\{x\}}$, so that $s_i^{\{x\}}$ represents the probability of the i-th token in the model's vocabulary given the input context x.

It is then a standard result of the softmax function that:

$$\frac{\partial}{\partial \theta_j} \log s_i^{\{x\}} = 1\{i = j\} - s_j^{\{x\}}. \tag{5}$$

Consider the case where y_w and y_l differ only at the first token, i.e., $m = 1$, and without loss of generality, assume that t_k takes vocabulary position 1. In this instance, for each term of Equation (4) where $k > 1$ we have[1]:

$$\nabla_{\theta_j} \log \pi_\theta(t_k|y_w^{<k}, x) - \nabla_{\theta_j} \log \pi_\theta(t_k|y_l^{<k}, x) = 1\{1 = j\} - s_j^{\{y_w^{<k}, x\}} - (1\{1 = j\} - s_j^{\{y_l^{<k}, x\}})$$

$$= s_j^{\{y_l^{<k}, x\}} - s_j^{\{y_w^{<k}, x\}}. \tag{6}$$

As we typically run DPO after SFT the model is likely to be reasonably well optimised, so we should have $s_j^{\{y_w^{<k}, x\}} \le s_j^{\{y_l^{<k}, x\}}$ for $j \neq 1$ and $s_1^{\{y_w^{<k}, x\}} \ge s_1^{\{y_l^{<k}, x\}}$. Therefore, while this analysis only extends to gradients with respect to the logits, we see that the gradient vector is decreasing in the correct logit dimension and increasing in the wrong logit dimensions. <mark>In particular, this derivation suggests that under DPO, all tokens that follow a difference at m at any point should have reduced probability of emitting the correct token when compared to π_{ref}.</mark>

B.2 Derivation for DPOP

We can follow a similar line of reasoning for calculating $\nabla_\theta \mathcal{L}_{DPOP}$ with respect to its logits which we denote again by θ_j.

Again taking token position k for illustrative purposes and assuming that token t_k has vocabulary position i, in the case when $\pi_{\text{ratio}}(y|x) < 1$,

$$\nabla_{\theta_j} [\log \pi_\theta(t_k|y_w^{<k}, x) - \log \pi_\theta(t_k|y_l^{<k}, x) - \lambda \max(0, \log \frac{\pi_{\text{ref}}(t_k|y_w^{<k}, x)}{\pi_\theta(t_k|y_w^{<k}, x)})]$$

$$= \nabla_{\theta_j} [\log \pi_\theta(t_k|y_w^{<k}, x) - \log \pi_\theta(t_k|y_l^{<k}, x) - \lambda \log \frac{\pi_{\text{ref}}(t_k|y_w^{<k}, x)}{\pi_\theta(t_k|y_w^{<k}, x)}]$$

$$= \nabla_{\theta_j} [\log \pi_\theta(t_k|y_w^{<k}, x) - \log \pi_\theta(t_k|y_l^{<k}, x) + \lambda \log \pi_\theta(t_k|y_w^{<k}, x)]$$

$$= (1 + \lambda) \nabla_{\theta_j} \log \pi_\theta(t_k|y_w^{<k}, x) - \nabla_{\theta_j} \log \pi_\theta(t_k|y_l^{<k}, x)$$

$$= (1 + \lambda)(1\{i = j\} - s_j^{\{y_w^{<k}, x\}}) - (1\{i = j\} - s_j^{\{y_l^{<k}, x\}})$$

$$= \begin{cases} \lambda(1 - s_j^{\{y_w^{<k}, x\}}) + s_j^{\{y_l^{<k}, x\}} - s_j^{\{y_w^{<k}, x\}} & i = j \\ -(\lambda + 1)s_j^{\{y_w^{<k}, x\}} + s_j^{\{y_l^{<k}, x\}} & i \neq j \end{cases}$$

---

[1] Note that the preceding negative sign is dropped as it is the gradient of a loss function and so during optimisation we take steps in the opposite direction, so as to minimise the loss.

Since $s_j^{\{y_w^{<k}, x\}} \leq 1$, for the case $i = j$, the gradient is guaranteed to be positive for a large enough choice of λ. Similarly, for the case $i \neq j$, the gradient is guaranteed to be negative for a large enough λ (as long as $s_j^{\{y_w^{<k}, x\}} > 0$). This therefore fixes the issue in DPO from Section 3.

In the case when $\pi_{\text{ratio}}(y|x) \geq 1$, then we have the standard gradient from $\mathcal{L}_{\text{DPO}}$.

C   Motivation: Contrastive Loss

While the main motivation for DPOP is to avoid the failure mode described in Section 3, we also note its connection to *contrastive loss*. Contrastive learning is widely used [Wang and Liu, 2021, Wang and Isola, 2020, Saunshi et al., 2019, Oord et al., 2018, Chen et al., 2020, He et al., 2020], often for embedding learning applications. The contrastive loss formulation typically includes two main terms: one encouraging the proximity of analogous inputs, the other encouraging the divergence of distinct classifiable data.

Moreover, the introduction of a margin appended to one of these terms often ensures a more stable training process. This margin serves as an indicator of indifference to point displacement once a specific value threshold is exceeded. The margin, when attached to the similar points term, establishes a minimum threshold beyond which we do not care about pulling the points closer. Alternatively, if added on the dissimilar points term, the margin sets a maximum threshold.

We show that the DPO loss is structured such that learning the probabilities during DPO training are equivalent to learning the embeddings in a contrastive loss formulation. However, the standard DPO only uses the term computing distance between dissimilar points, and does not include the similar points term or the margin. Consequently, it is predictable that traditional DPO's inefficiencies mirror the known shortcomings of contrastive training when one constituent term is absent. DPOP, our refined DPO formulation, fixes this by adding the absent term and the margin.

Contrastive loss is defined in Hadsell et al. [2006]. If we keep the margin in the similar points terms, it can be written as follows:

$$\mathcal{L}_{\text{Cont}} = - \sum_{\forall (i,j) \epsilon \mathcal{P}_d} \mathcal{D}(y_i, y_j) + \lambda \sum_{\forall (i,j) \epsilon \mathcal{P}_s} \min(\mathcal{D}(y_i, y_j) - m, 0)$$

Recall that the standard DPO loss (Equation (1)) is as follows:

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

Say we designate an embedding function $\mathcal{H}$:

$$\mathcal{H}(y|x) = \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}.$$

And we define a distance function $\mathcal{D}$ as follows:

$$\mathcal{D}(p_i, p_j) = \log[\mathcal{H}(p_i)] - \log[\mathcal{H}(p_j)].$$

The standard DPO only has the dissimilar points term under the analogy of the contrastive loss formulation. For more robust training we accommodate for the similar embeddings term. We use the concept of anchor points or embeddings for both positive and negative points as in triplet loss Schroff et al. [2015]. These points are known ideal embeddings we want our points to achieve. They carry probabilities of 1 and 0 respectively in our equivalence depending on whether they are preferred or dispreferred samples.

$$\mathcal{H}_p^*(y|x) = \frac{1}{\pi_{\text{ref}}(y|x)}$$

$$\mathcal{H}_n^*(y|x) = \left.\frac{\epsilon}{\pi_{\text{ref}}(y|x)}\right|_{\epsilon \to 0}$$

The DPOP loss can thus be formulated as:

$$\mathcal{L}_{\text{DPOP}} = [\log(\mathcal{H}(y_w|x)) - \log(\mathcal{H}(y_l|x))] + \lambda \left[\min\left(\log(\mathcal{H}_p^*(y_w|x)) - \log(\mathcal{H}(y_w|x)) - m, 0 \right) \right.$$

$$\left. + \min\left(\log(\mathcal{H}_n^*(y_l|x)) - \log(\mathcal{H}(y_l|x)) - m, 0 \right) \right]$$

$$= \left[\log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \log\frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)} \right] + \lambda \left[\min\left(\log\frac{1}{\pi_{\text{ref}}(y_w|x)} - \log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - m, 0 \right) \right]$$

$$+ \lambda \left[\min\left(\log\frac{\epsilon}{\pi_{\text{ref}}(y_l|x)} - \log\frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)} - m, 0 \right) \right]$$

If we set the margin $m = \log\frac{1}{\pi_{\text{ref}}(y_w|x)}$, the second term is:

$$= -\lambda \left[\max\left(\log\frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)}, 0 \right) \right]$$

This choice of margin is mathematically equivalent to choosing a threshold which ensures the similarity term only contributes to the loss when the learned model performs worse on the preferred response than the base model.

We can ignore the third term during training for two primary reasons. First, it is trying to push the log probability of negative samples to negative infinity which may be unstable during training. Second, in essence, it negatively impacts the likelihood of the negative samples. However, given our objective of extending the distance between positive and negative samples without diminishing the likelihood of positives, sacrificing this signal is acceptable. In the worst-case scenario, while the probability of negatives may be increased, the consequent increase in the probability of positives is ensured by the first two terms, thus rendering the sacrifice acceptable.

D   Details of Datasets

In this section, we give more details and examples of the three paired preference versions of datasets we release. We release all datasets under the Apache 2.0 License.

MetaMath   GSM8K [Cobbe et al., 2021] is a dataset of diverse grade school maths word problems, which has been commonly adopted as a measure of the math and reasoning skills of LLMs [Chowdhery et al., 2023, Touvron et al., 2023b,a, Beeching et al., 2023, Gao et al., 2021]. The MetaMath dataset is partitioned into queries and responses, where the query is a question involving mathematical calculation or reasoning, and the response is a logical series of steps and calculations that culminate in a final answer. To construct our paired-preference version of MetaMath, we take the queries as prompts x and the responses as the preferred completions y_w. We create y_l by modifying the response by randomly corrupting one of the results of an intermediate calculation so that it is incorrect; however, we leave the answer untouched.

An example is as follows: x: "What is the total cost of purchasing equipment for all sixteen players on the football team, considering that each player requires a \$25 jersey, a \$15.20 pair of shorts, and a pair of socks priced at \$6.80?" y_w: "Each player requires a \$25 jersey, a \$15.20 pair of shorts, and a pair of socks priced at \$6.80. So the total cost for each player is \$25 + \$15.20 + \$6.80 = \$**47**. Since there are sixteen players on the football team, the total cost for all of them is 16 * \$47 = \$752. #### 752 The answer is: 752" y_l: "Each player requires a \$25 jersey, a \$15.20 pair of shorts, and a pair of socks priced at \$6.80. So the total cost for each player is \$25 + \$15.20 + \$6.80 = \$**52**. Since there are sixteen players on the football team, the total cost for all of them is 16 * \$47 = \$752. #### 752 The answer is: 752"

The dataset contains 393 999 training examples and 1 000 evaluation examples. Our motivation in building this dataset is to align models towards being precise in intermediate calculations. This dataset has low edit distance – the normalised edit distance is approximately 6.5%.

ARC ARC [Clark et al., 2018] is a dataset that tests the level of understanding of science at approximately grade-school level. We focus specifically on the 'Challenge' subsection of ARC, the more difficult of the two subsections, which has been widely adopted as a measure of LLM reasoning and world understanding [Chowdhery et al., 2023, Touvron et al., 2023b,a, Beeching et al., 2023, Gao et al., 2021, Cobbe et al., 2021]. We create a paired preference-ranked dataset from the train split of ARC-Challenge. The dataset is partitioned into questions which we take as our prompts x, and four choices of responses to each question of which only one is the correct answer. The correct response is taken as y_w and the incorrect responses are taken to be y_l; as there are three incorrect responses for every prompt, we repeat y_w multiple times for each prompt. The dataset contains 3357 training examples and 895 evaluation examples. This dataset has a high normalised edit distance of approximately 90%.

HellaSwag Finally, we consider the HellaSwag dataset [Zellers et al., 2019], a dataset containing common-sense inference questions known to be hard for LLMs. An example prompt is "Then, the man writes over the snow covering the window of a car, and a woman wearing winter clothes smiles. then" And the potential completions are [", the man adds wax to the windshield and cuts it.", ", a person board a ski lift, while two men supporting the head of the person wearing winter clothes snow as the we girls sled.", ", the man puts on a christmas coat, knitted with netting.", ", the man continues removing the snow on his car."] The dataset contains 119 715 training and 30 126 evaluation examples.

E Additional Experiments and Details

E.1 Additional Training Details

No hyperparameter tuning was done when creating Smaug-34B or Smaug-72B. DPOP has two hyperparameters, β and λ. We chose $\beta = 0.3$, similar to prior work [Rafailov et al., 2023], and we chose $\lambda = 50$ without trying other values. It is possible that even better performance can be achieved, e.g., with a different value of λ.

Here, we give the licenses of all models used to train our Smaug-series of models.

Smaug-7B started from Llama 2-chat [Touvron et al., 2023b]. Therefore, we release it under the Llama 2 license (https://ai.meta.com/llama/license/).

Smaug-34B started from Bagel-34B-v0.2 [Durbin, 2024a], which itself is a SFT version of Yi-34B-200k [01.AI, 2024]. Therefore, we release Smaug-34B under the Yi Series Models Community License Agreement (https://github.com/01-ai/Yi/blob/main/MODEL_LICENSE_AGREEMENT.txt).
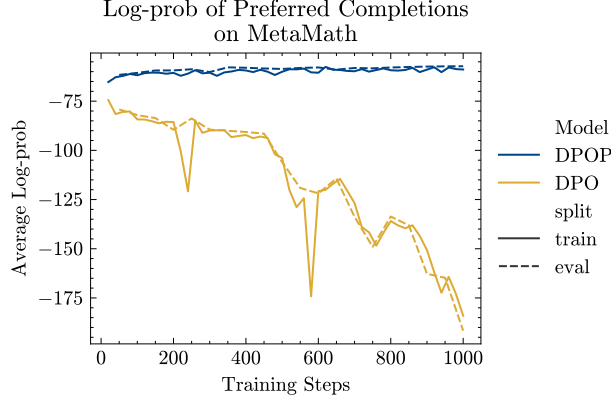
Figure 5: Average log-probs for preferred completions of DPOP and DPO on the MetaMath dataset showing the failure mode of DPO.
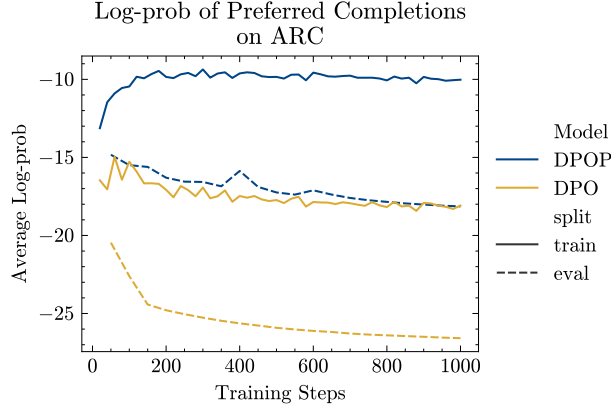

Figure 6: Average log-probs for preferred completions of DPOP and DPO on the ARC dataset showing the failure mode of DPO.

Smaug-72B started from MoMo-72b-lora-1.8.7-DPO [Moreh, 2024], which itself is a fine-tune of Qwen-72B [Bai et al., 2023]. Therefore, we release Smaug-72B under the Qwen license (https://github.com/QwenLM/Qwen/blob/main/Tongyi%20Qianwen%20LICENSE%20AGREEMENT).

E.2   Additional Results

Log-probabilities of preferred completions   In Figure 5, we show the log-probabilities of the preferred completion of the train and eval sets during training on MetaMath. We plot the log-probabilities in more granularity than in Figure 4. We confirm our theoretical insights from Section 4 – the log-probabilities of the preferred completion drop substantially in DPO, whereas they increase for DPOP – across both the train and eval sets. For ARC, we see in Figure 6 that DPOP maintains high train-set log-probs, while both the train and eval set log-probs decrease for DPO. Notably, even though eval set log-probs do decrease for DPOP, they are still higher than the train set log-probs of DPO.

Table 2: Evaluation of the top models according to MT-Bench and MMLU.

Model	MT-bench	MMLU	Organization	License
GPT-4-1106-preview	**9.32**		OpenAI	Proprietary
GPT-4-0314	8.96	**86.4**	OpenAI	Proprietary
GPT-4-0613	9.18		OpenAI	Proprietary
Mistral Medium	8.61	75.3	Mistral	Proprietary
Claude-1	7.9	77	Anthropic	Proprietary
Claude-2.0	8.06	78.5	Anthropic	Proprietary
Gemini Pro (Dev API)		71.8	Google	Proprietary
Claude-2.1	8.18		Anthropic	Proprietary
GPT-3.5-Turbo-0613	8.39		OpenAI	Proprietary
Mixtral-8x7b-Instruct-v0.1	8.3	70.6	Mistral	Apache 2.0
Yi-34B-Chat		73.5	01 AI	Yi License
Gemini Pro		71.8	Google	Proprietary
Claude-Instant-1	7.85	73.4	Anthropic	Proprietary
GPT-3.5-Turbo-0314	7.94	70	OpenAI	Proprietary
WizardLM-70B-v1.0	7.71	63.7	Microsoft	Llama 2 Community
Tulu-2-DPO-70B	7.89		AllenAI/UW	AI2 ImpACT Low-risk
Vicuna-33B	7.12	59.2	LMSYS	Non-commercial
Starling-LM-7B-alpha	8.09	63.9	UC Berkeley	CC-BY-NC-4.0
Smaug-72B	7.76	77.15	Abacus.AI	tongyi-qianwen-license-agreement

Additional tables   In Table 4, we give an extension of Table 1 (whose experimental details are in Section 6). In Table 5, we give the same table, except for models of size 34B or lower.

MT-Bench   Next, we evaluate using MT-Bench [Zheng et al., 2023], a challenging benchmark that uses GPT-4 [OpenAI, 2023] to score candidate model responses across eight different categories of performance. As shown in other works [Zheng et al., 2023, Rafailov et al., 2023], strong LLMs such as GPT-4 show good agreement with human preferences. We note that MT-Bench questions were not included in our fine-tuning datasets in any way. We run MT-Bench with the Llama-2 conversation template [Touvron et al., 2023b]. See Appendix Table 2 for a comparison with state-of-the-art LLMs according to Arena Elo as of March 2024. Smaug-72B achieves the top MMLU score and third-best MT-bench score out of the open-source models. In Appendix E, we give examples of Smaug-72B completions to MT-Bench questions.

E.3   Contamination Checker

Since LLMs train on wide swaths of the internet, data contamination, i.e., evaluating on examples that are included in the training data, is a growing concern. Data contamination remains notoriously challenging to measure and mitigate, even with partial attempts like controlled experimentation of training data, canary strings, or embedding similarities [Roberts et al., 2024, Jain et al., 2024, bench authors, 2023].

While there is no perfect tool to check for data contamination, we use an open-source contamination checker (https://github.com/swj0419/detect-pretrain-code-contamination) to compare the contamination of our model to other open-source models - see Table 3. On ARC, TruthfulQA, and GSM8K, we

Model	Smaug-72B	MoMo-72B-lora-1.8.7-DPO	Llama-2-70B
ARC	0.20	0.20	0.22
TruthfulQA	0.45	0.39	0.51
GSM8K	1.00	1.00	0.89

Table 3: Comparison of contamination values across models, according to https://github.com/swj0419/detect-pretrain-code-contamination. A score above 0.85 indicates likely contamination.

Table 4: (Extension of Table 1.) Evaluation of the top open-weight models on the HuggingFace Open LLM Leaderboard as of March 2024. Our 72B model achieves an average accuracy of 80.48%, becoming the first open-source LLM to surpass an average accuracy of 80% and improving by nearly 2% over the second-best open-source model (other than a fine-tune of our own model).

Model	Avg.	ARC	HellaSwag	MMLU	TruthfulQA	Winogrande	GSM8K
Smaug-72B (Ours)	**80.48**	**76.02**	89.27	77.15	**76.67**	85.08	**78.70**
MoMo-72B-lora-1.8.7-DPO	78.55	70.82	85.96	77.13	74.71	84.06	78.62
TomGrc_FusionNet_34Bx2_MoE_v0.1_DPO_f16	77.91	74.06	86.74	76.65	72.24	83.35	74.45
TomGrc_FusionNet_34Bx2_MoE_v0.1_full_linear_DPO	77.52	74.06	86.67	76.69	71.32	83.43	72.93
Truthful_DPO_TomGrc_FusionNet_7Bx2_MoE_13B	77.44	74.91	**89.30**	64.67	78.02	**88.24**	69.52
CCK_Asura_v1	77.43	73.89	89.07	75.44	71.75	86.35	68.08
FusionNet_34Bx2_MoE_v0.1	77.38	73.72	86.46	76.72	71.01	83.35	73.01
MoMo-72B-lora-1.8.6-DPO	77.29	70.14	86.03	**77.40**	69.00	84.37	76.80
Smaug-34B-v0.1 (Ours)	77.29	74.23	86.76	76.66	70.22	83.66	72.18
Truthful_DPO_TomGrc_FusionNet_34Bx2_MoE	77.28	72.87	86.52	76.96	73.28	83.19	70.89
DARE_TIES_13B	77.10	74.32	89.5	64.47	78.66	88.08	67.55
13B_MATH_DPO	77.08	74.66	89.51	64.53	78.63	88.08	67.10
FusionNet_34Bx2_MoE	77.07	72.95	86.22	77.05	71.31	83.98	70.89
MoE_13B_DPO	77.05	74.32	89.39	64.48	78.47	88.00	67.63
4bit_quant_TomGrc_FusionNet_34Bx2_MoE_v0.1_DPO	76.95	73.21	86.11	75.44	72.78	82.95	71.19
MixTAO-7Bx2-MoE-Instruct-v7.0	76.55	74.23	89.37	64.54	74.26	87.77	69.14
Truthful_DPO_cloudyu_Mixtral_34Bx2_MoE_60B0	76.48	71.25	85.24	77.28	66.74	84.29	74.07
MoMo-72B-lora-1.8.4-DPO	76.23	69.62	85.35	77.33	64.64	84.14	76.27
FusionNet_7Bx2_MoE_v0.1	76.16	74.06	88.90	65.00	71.20	87.53	70.28
MBX-7B-v3-DPO	76.13	73.55	89.11	64.91	74.00	85.56	69.67

find that Smaug-72B achieves scores that are similar to MoMo-72B-lora-1.8.7-DPO (the starting point of Smaug-72B), as well as Llama-2-70B.

F Example Completions

In this section, we give example completions by Smaug-72B for questions in MT-Bench [Zheng et al., 2023]. Note that these are not cherry-picked – they include examples of both good and bad completions.

Table 5: Evaluation of the top open-weight models on the HuggingFace Open LLM Leaderboard for models less than 35B parameters as of March 2024. Our 34B model achieves best-in-its-class performance compared to other models of similar or smaller size.

Model	Size	Avg.	ARC	HellaSwag	MMLU	TruthfulQA	Winogrande	GSM8K
Smaug-34B-v0.1 (Ours)	<35B	**77.29**	74.23	86.76	**76.66**	70.22	83.66	**72.18**
DARE_TIES_13B	<35B	77.10	74.32	89.50	64.47	**78.66**	**88.08**	67.55
13B_MATH_DPO	<35B	77.08	**74.66**	**89.51**	64.53	78.63	**88.08**	67.10
MoE_13B_DPO	<35B	77.05	74.32	89.39	64.48	78.47	88.00	67.63
4bit_quant_TomGrc_FusionNet_34Bx2_MoE_v0.1_DPO	<35B	76.95	73.21	86.11	75.44	72.78	82.95	71.19

Example 1

Question: Imagine you are writing a blog post comparing two popular smartphone models. Develop an outline for the blog post, including key points and subheadings to effectively compare and contrast the features, performance, and user experience of the two models. Please answer in fewer than 200 words.

Outline:

1. Introduction to the two smartphone models being compared (mention brand names and models)
2. Design and Build Quality
   - Comparison of dimensions, weight, materials used, and overall aesthetic appeal
   - Water and dust resistance, if applicable
3. Display and Screen Quality
   - Comparison of screen technology (OLED, IPS LCD, etc.)
   - Display resolution, refresh rate, and brightness levels
4. Performance and Hardware
   - Processor, RAM, and storage configurations
   - Comparison of benchmark scores (e.g., AnTuTu, Geekbench)
5. Camera System and Image Quality
   - Comparison of primary, ultra-wide, telephoto, and macro lenses (if available)
   - Comparison of low-light performance and image processing capabilities
6. Battery Life and Charging
   - Battery capacity and estimated usage time
   - Charging technologies (wired, wireless, fast charging, reverse charging)
7. Software and User Experience
   - Comparison of operating systems (iOS vs. Android) and their features
   - User interface, customization options, and app ecosystem
8. Additional Features and Extras
   - Comparison of biometric authentication (fingerprint, face unlock, etc.)
   - Additional features like NFC, 5G connectivity, and audio quality
9. Pricing and Value for Money
   - Comparison of retail prices and current offers
   - Assessment of overall value considering the features and performance
10. Conclusion and recommendation for potential buyers based on their preferences

Example 2

Question: In a survey conducted at a local high school, preferences for a new school color were measured: 58% of students liked the color blue, 45% preferred green, and 22% liked both colors. If we randomly pick a student from the school, what's the probability that they would like neither blue nor green?

To find the probability that a randomly picked student would like neither blue nor green, we need to find the percentage of students who like only one color or neither color.
From the given information, we know:

- 58% like blue
- 45% prefer green
- 22% like both colors

To find the percentage of students who like only one color, we can subtract the percentage of students who like both colors from the sum of students who like blue and green:

- 58% (blue) + 45% (green) - 22% (both) = 83%

Now, to find the probability of a student liking neither blue nor green, we subtract the percentage of students who like at least one color from 100%:

- 100% - 83% = 17%

So, the probability that a randomly picked student would like neither blue nor green is 17%.

Example 3

Question: Implement a program to find the common elements in two arrays without using any extra data structures.

Finding common elements in two arrays without using extra data structures can be achieved by using a simple algorithm. Here's a Python implementation of this problem:

```python
def find_common_elements(arr1, arr2):
    common_elements = []
    for num1 in arr1:
        if num1 in arr2 and num1 not in common_elements:
            common_elements.append(num1)
    return common_elements

# Example usage:
array1 = [1, 2, 3, 4, 5]
array2 = [4, 5, 6, 7, 8]
print(find_common_elements(array1, array2)) # Output: [4, 5]
```

This program iterates through the elements of the first array (arr1) and checks if each element is present in the second array (arr2). If it is, and the element hasn't been added to the common_elements list yet, it is appended. The function then returns the list of common elements.

Example 4

Question: Identify the named entities (people, organizations, locations) mentioned in the given news article. Please generate a JSON dictionary that lists the named entities in three separate groups based on their entity types. The key is the type of entity and the value is a list of strings.

Yesterday, Adamson Emerson, the CEO of Faraday, and Dieter Zetsche, the CEO of Daimler AG, announced plans to build a new Gigafactory in Berlin. The facility will be a joint venture between Faraday and Daimler, producing electric vehicles and battery packs for both companies, creating thousands of job opportunities in the region. Emerson and Zetsche stated that the strategic location of Berlin, coupled with its skilled workforce and strong infrastructure, makes it an ideal choice for expansion. The new Gigafactory aims to meet the growing demand for electric vehicles in Europe and contribute to a sustainable future. Volkswagen CEO Herbert Diess welcomed the news, saying greater collaboration will, benefit the auto industry's transition to e-mobility.

```
{
    "people": ["Adamson␣Emerson", "Dieter␣Zetsche", "Herbert␣Diess"],
    "organizations": ["Faraday", "Daimler␣AG", "Volkswagen"],
    "locations": ["Berlin", "Europe"]
}
```