GAW-PO: PREFERENCE OPTIMIZATION WITH GRADIENT-ALIGNED TOKEN WEIGHTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Most preference optimization methods, such as Direct Preference Optimization (DPO), apply preference supervision at the response level, although autoregressive language models are optimized token by token. As a result, all tokens in a rejected response contribute to the negative training signal, including tokens that may encode behavior that is useful for the preferred response. We introduce GAW-PO, a gradient-aligned token reweighting method for DPO that estimates, for each rejected token, whether penalizing it would interfere with the preferred update directions. Tokens whose gradients are strongly aligned with the preferred behavior receive a weaker negative contribution, while conflicting tokens retain a stronger penalty. Our method achieves the highest average performance among the evaluated preference-optimization methods, improving by 0.97 points over standard DPO and 0.65 points over the strongest competing baseline across 11 benchmarks spanning mathematics, reasoning, coding, and question answering. We further show that gradient-aligned weighting is substantially more robust to aggressive preference optimization: as the DPO regularization parameter β decreases, standard DPO degrades sharply, whereas GAW-PO continues to improve. These results suggest that accounting for the interaction between rejected-token updates and preferred behavior provides an effective form of token-level credit assignment for preference optimization.

1 INTRODUCTION

Preference optimization has become a standard approach for aligning Large Language Models (LLMs) with human preferences. Among these methods, Direct Preference Optimization (DPO) (Rafailov et al., 2023) is widely used due to its simplicity and effectiveness. Given a prompt, a chosen (preferred) response, and a rejected response, DPO trains the model to increase the relative preference for the chosen completion over the rejected one with respect to a reference policy. However, the supervision available to DPO is defined at the response level, while autoregressive language models operate at the token level. As a consequence, all tokens in a rejected response contribute to the optimization signal, even though only a small subset of them may be responsible for the response being considered of lesser quality.

This mismatch can lead to undesirable or at least inefficient updates during preference optimization. A rejected completion is rarely incorrect in its entirety: it may contain valid facts, correct intermediate calculations, appropriate formatting, or linguistic patterns that are also useful for generating the preferred response. Nevertheless, standard DPO does not explicitly distinguish these locally correct components from the tokens that actually determine the preference. Reducing the likelihood of the rejected response can therefore also suppress behaviors that should ideally be preserved. Figure 1 illustrates a simple example of this effect. More generally, the DPO objective constrains the relative preference between the chosen and rejected completions, but does not require the absolute likelihood of the chosen response to increase. Indeed, several works have observed that the likelihood of preferred responses can decrease during DPO training even while the preference objective improves (Pal et al., 2024; Pang et al., 2024). This behavior suggests that the optimization dynamics of preferred and rejected responses can be strongly coupled (Razin et al., 2025).

This limitation can be addressed by introducing finer-grained supervision at the token level. Recent methods have proposed token-level preference objectives (Zeng et al., 2024) or weighting

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

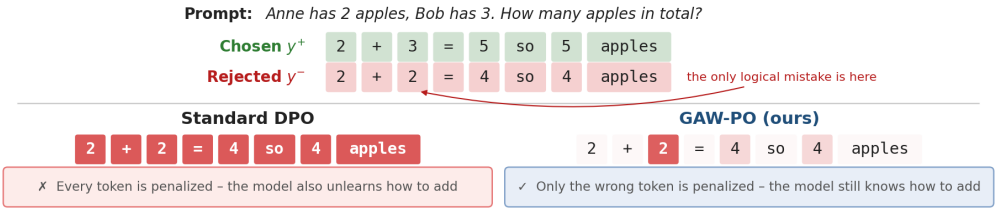

Figure 1: Standard DPO penalizes every token of the rejected response, including the arithmetic; our method concentrates the penalty on the token that conflicts with the preferred response.

schemes (Liu et al., 2025; Yang et al., 2026; Li et al., 2025) that avoid treating every token in a response as equally informative. These methods primarily aim to estimate how important or informative an individual token is. Existing methods alone, however, do not fully characterize the consequence of penalizing that token during optimization.

Conversely, we consider the effect that suppressing a rejected token would have on the model's ability to represent the preferred response. Our starting point is the observation that training samples interact through their parameter updates. This idea has a long history in machine learning, most prominently in work on influence functions (Koh & Liang, 2017) and gradient-based influence estimation (Pruthi et al., 2020). In these approaches, the relationship between training examples can be characterized by comparing the parameter updates they induce. Examples with similar gradient directions tend to modify the model in compatible ways, while opposing gradients indicate conflicting training signals. Related ideas have also been applied to LLMs, for example for selecting instruction-tuning or reasoning data based on gradient similarity (Xia et al., 2024; Jung et al., 2026).

We apply this perspective to token-level preference optimization. Rather than assigning a weight to a rejected token based only on its probability, loss, or gradient magnitude, we investigate how the update induced by penalizing that token interacts with the update supported by the preferred response. If a rejected token induces a parameter update that is strongly aligned with the one associated with the preferred completion, suppressing that token is likely to interfere with behavior that is also useful for modeling the preferred response. Such a token should therefore receive a weaker negative signal. Conversely, if the token induces an update that is dissimilar to, or conflicts with, the preferred-response update, penalizing it is less likely to damage the behavior encouraged by the chosen completion and can therefore receive a stronger negative signal. Based on this principle, we introduce GAW-PO, a gradient-aligned token reweighting method for DPO that explicitly accounts for interference between rejected-token updates and preferred update directions.

Our main contributions are: **(1)** We propose a gradient-aligned token weighting objective for DPO that selectively downweights rejected tokens whose penalization would interfere with behavior supported by preferred responses. **(2)** We show that our formulation improves average downstream performance over standard DPO and competing methods, while remaining stable under substantially more aggressive optimization regimes in which standard DPO degrades sharply. We open-source our implementation, gradient-weight computation pipeline, training recipes, filtered preference data, and evaluation setup to facilitate reproducibility and further study: `https://anonymous.4open.science/r/Weighted-DPO-Supplementary-8441/`

2 Related Work

Preference optimization. DPO (Rafailov et al., 2023) replaced the reward model and reinforcement learning stage of RLHF (Christiano et al., 2017; Ouyang et al., 2022) with a single classification loss on preference pairs. Most of its variants keep the entire response as the unit of optimization: IPO (Azar et al., 2024) changes the loss to limit overfitting, KTO (Ethayarajh et al., 2024) learns from unpaired feedback, and SimPO (Meng et al., 2024b) replaces the reference model with a length-normalized reward. A known side effect of this response-level objective is that the likelihood of the chosen response can decrease during training, in particular when the two responses differ in only a few tokens (Pal et al., 2024; Pang et al., 2024; Razin et al., 2025). The previously proposed solutions

also act on whole responses, either by adding a likelihood term on the chosen response (Pal et al., 2024; Pang et al., 2024) or by filtering out pairs with similar representations (Razin et al., 2025). Our method keeps every pair and changes only how strongly each rejected token is penalized.

Token- and step-level credit assignment. Finer-grained methods differ mainly in the source of the token-level signal. TDPO (Zeng et al., 2024) rewrites the Bradley–Terry objective in terms of token-level advantages but still weights all tokens equally, while TIS-DPO (Liu et al., 2025) takes token weights from the probability gap between two contrastive LLMs, which have to be obtained by prompting or trained separately. For reasoning, Step-DPO (Lai et al., 2024) moves from tokens to steps: it finds the first incorrect step of a wrong solution and contrasts a correct and an incorrect continuation of the same correct prefix. This leaves the shared steps unpenalized, but requires locating the error, manually or using a strong LLM judge, and resampling continuations. Similar to our approach, TI-DPO (Yang et al., 2026) also relies on gradients, but computes them with respect to the input embeddings and weights each token by the gradient norm, which measures how strongly the output depends on the token and not whether penalizing it would work against the chosen response.

AAO (Li et al., 2025) is motivated by the same problem as our method: DPO pushes the same content up in the chosen response and down in the rejected one. It compares each token with the tokens of the other response using the model's input embeddings, and assigns lower weights to tokens that have a close match. However, input embeddings ignore context, so a rejected token is treated as shared whenever the same token appears anywhere in the chosen response. In Figure 1, this includes the incorrect operand "2", which also appears as the first operand of the chosen response.

Gradient interaction between training signals. GAW-PO measures the interaction between rejected-token and chosen-response updates directly. Following the use of gradient inner products to relate training examples in influence estimation and data selection (Koh & Liang, 2017; Pruthi et al., 2020; Xia et al., 2024), we score each rejected token by the inner product between its parameter gradient and the gradients of chosen responses, from its own pair and averaged over the training dataset. Unlike the TI-DPO and AAO scores, this score is signed and depends on response context, and it is computed once before training without an auxiliary model. The closest work in terms of intent is NTHR (Deng et al., 2026), which reduces the penalty on tokens of incorrect responses in GRPO by approximating their gradient interactions through hidden-state and prediction-error alignment. Most token-level methods above were evaluated on dialogue, safety or summarization with LLM judges, or on a single domain, while we evaluate on 11 benchmarks across mathematics, reasoning, code, and question answering.

3 METHOD

3.1 PRELIMINARIES

We consider a preference dataset $\mathcal{D} = (x, y^+, y^-)$, where x denotes the prompt, y^+ the chosen response, and y^- the rejected response. Direct Preference Optimization (DPO) (Rafailov et al., 2023) optimizes a policy π_θ relative to a fixed reference policy π_{ref} according to the following loss:

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

where β penalizes the deviation from the reference policy. The objective therefore encourages a larger margin between the policy's relative likelihood of the chosen and rejected responses.

For an autoregressive language model, the log-likelihood of the rejected response decomposes as a sum over its tokens. Standard DPO implicitly assigns the same contribution to all tokens in this sum, regardless of whether a particular token is actually responsible for the response being rejected. We instead associate each rejected token y_t^- with a weight w_t and replace the rejected sequence-level log-ratio with a weighted sum of per-token log-ratios:

$$\log\frac{\pi_\theta(y^- \mid x)}{\pi_{\text{ref}}(y^- \mid x)} \quad\longrightarrow\quad \sum_{t=1}^{|y^-|} w_t\left[\log\pi_\theta(y_t^- \mid x, y_{<t}^-) - \log\pi_{\text{ref}}(y_t^- \mid x, y_{<t}^-)\right]. \tag{2}$$

The resulting weights control how strongly individual rejected tokens contribute to the negative preference signal. Tokens assigned small weights are only weakly penalized, whereas tokens assigned large weights retain a stronger contribution to the DPO objective. We constrain each weight to lie in $[0, 1]$, where $w_t = 1$ recovers the contribution assigned to that token under standard DPO. Thus, our method does not amplify the penalty of any rejected token beyond the standard DPO objective; instead, it selectively downweights tokens whose penalization is estimated to interfere with preferred behavior. The relation with the standard DPO objective is detailed in Appendix A.1.

3.2 Gradient-Based Token Weights

Our goal is to assign gradient-based token weights according to the effect that penalizing each rejected token would have on learning the chosen response. To characterize this interaction, we consider the parameter-space update induced by each token. For a rejected token y_t^-, we compute the gradient of its conditional log-likelihood with respect to the model parameters:

$$g_t^- = \nabla_\theta \log \pi_\theta(y_t^- \mid x, y_{<t}^-). \tag{3}$$

This gradient represents the direction in parameter space that would increase the probability of the rejected token. Consequently, penalizing the token induces an update in the opposite direction. We compare this token-level gradient with a gradient representing the behavior encouraged by the chosen response. In the simplest setting, this gradient is computed using the complete chosen response:

$$g^+ = \nabla_\theta \log \pi_\theta(y^+ \mid x). \tag{4}$$

For each rejected token, we compute two similarity scores against different reference directions. As the first source, we compare its gradient g_t^- with the gradient associated with its corresponding chosen response, denoted by g_{pair}^+. This captures whether penalizing the rejected token would interfere with learning the preferred response from the same preference pair. In terms of the second source, we compare g_t^- with an aggregate chosen gradient averaged across all $N = |\mathcal{D}|$ preference pairs, $g_{\text{global}}^+ = \frac{1}{N} \sum_{i=1}^{N} g_{\text{pair},i}^+$. This second similarity captures whether penalizing the token would interfere with parameter directions that are consistently supported by preferred responses more broadly. We denote these two similarities as

$$s_t^{\text{pair}} = \text{sim}(g_t^-, g_{\text{pair}}^+) \qquad \text{and} \qquad s_t^{\text{global}} = \text{sim}(g_t^-, g_{\text{global}}^+). \tag{5}$$

Strong alignment with either source is sufficient to indicate that penalizing the token may suppress useful preferred behavior. As such, we take the maximum of the two similarities for each rejected token to ensure a conservative criterion:

$$s_t = \max\left(s_t^{\text{pair}}, s_t^{\text{global}}\right). \tag{6}$$

This choice reflects two complementary notions of interference. A rejected token receives a weaker penalty if suppressing it would conflict either with its paired chosen response or with behavior that is consistently supported by chosen responses across the dataset. If the resulting similarity is high, the rejected token and at least one of the chosen signals induce similar parameter updates. Penalizing that token would therefore move the model in the opposite direction and may interfere with preferred behavior, so we assign it a smaller weight. Conversely, when the rejected-token gradient is dissimilar to both its paired chosen gradient and the aggregate chosen gradient, suppressing the token is less likely to interfere with preferred behavior, and it receives a larger loss weight.

Finally, we transform s_t into a token weight $w_t \in [0, 1]$ using a monotonically decreasing function. Weights close to 1 approach the original contribution of the token under standard DPO, while smaller values progressively reduce its negative contribution. In this manner, our weighting mechanism protects both preference-specific information from the paired chosen response and more general preferred behavior across the entire dataset.

3.3 Experimental and Implementation Decisions

Computing a separate token gradient over all model parameters is prohibitively expensive for modern LLMs. We therefore measure gradient alignment in the parameter space of a rank-64 PiSSA-initialized adapter (Meng et al., 2024a) attached to the frozen base model. The adapter is used only as a *gradient probe*; the subsequent DPO stage performs full-model fine-tuning. Because PiSSA initializes the low-rank factors from the principal singular components of each weight matrix, the adapter gradients are projections of the full-parameter gradients onto these principal subspaces (see Appendix A.2). A property reported by Meng et al. (2024a) motivates this choice: the training loss and gradient-norm trajectories of PiSSA follow those of full-parameter fine-tuning closely, and do so consistently across ranks from 1 to 128 on several models. This indicates that gradients restricted to the principal subspaces preserve most of the full-parameter gradient signal.

To construct the chosen direction, we aggregate the token-level gradients of the chosen completion while compensating for differences in their magnitude. For cross-entropy, the gradient of $\log p_i^+$ with respect to the logits has L_1 norm $2(1 - p_i^+)$, where p_i^+ is the probability assigned to the target token. As such, tokens that the model already predicts confidently contribute close to nothing to an unweighted sum, and the aggregate direction is determined by the few tokens the model finds surprising. We instead weight each chosen-token gradient by the inverse of its logit-level L_1 norm,

$$g^+ = \sum_{i=1}^{T^+} \alpha_i g_i^+, \qquad \alpha_i = \frac{1}{2(1 - p_i^+)}, \tag{7}$$

with α_i treated as a constant (detached from the computational graph). The result approximates an unweighted average over token gradient directions rather than a magnitude-weighted one, which is the intended reference for the alignment scores: the direction should reflect what the chosen response as a whole supports, not what its most surprising tokens support. The derivation and the numerical stabilization are given in Appendix A.3.

We use the *dot product* to measure gradient alignment. The dot product is particularly convenient computationally: once g_t^- is obtained, its projections onto multiple reference directions can be computed directly, and only the resulting scalar products need to be stored. We therefore never materialize the full token-by-parameter gradient matrix. Rejected tokens are processed in chunks to further bound memory usage. More details can be found in Appendix A.4.

Finally, we convert the alignment scores into rejected-token weights. Since the raw dot products scale with the gradient norms and these vary considerably across examples, we normalize the scores of each rejected response by their largest absolute value and apply a sigmoid to the negated result:

$$s_t = \max\left(\left\langle g_t^-, g_{\text{pair}}^+ \right\rangle, \left\langle g_t^-, g_{\text{global}}^+ \right\rangle\right), \qquad w_t = \sigma\left(-\frac{s_t}{\max_{t'} |s_{t'}|}\right), \tag{8}$$

where the maximum runs over all tokens of the same rejected response.

The $\frac{s_t}{\max_{t'} |s_{t'}|}$ mapping is symmetric around 0 because the score is signed and both signs are informative: *positive* alignment marks a token whose suppression would interfere with preferred behavior, *negative* alignment marks a conflicting token whose suppression moves the model toward the preferred update, while score close to zero denote *orthogonal* tokens who are separated from conflicting ones rather than collapsed onto the same weight. The weights attenuate the rejected penalty at individual positions, and since no weight exceeds 1, no rejected token is penalized more strongly than under standard DPO. Highly aligned tokens receive smaller weights, while conflicting tokens receive weights closer to 1 and therefore retain more of the standard DPO penalty. Only rejected tokens are reweighted; chosen-token weights remain fixed to 1. Appendix D.1 shows the resulting weights on concrete examples.

3.4 Algorithm and Training Procedure

Algorithm 1 summarizes the complete procedure. We first precompute rejected-token weights using gradient alignment in the frozen PiSSA probe space, then keep these weights fixed during full-model DPO training. We use $G^+(x, y^+)$ to denote the chosen-response gradient direction defined

in Equation 7; each rejected token is compared against the pair-specific and dataset-level chosen directions, with stronger alignment yielding a smaller DPO penalty.

Algorithm 1 GAW-PO: Gradient-Aligned Token Reweighting for DPO

Require: Preference dataset $\mathcal{D} = \{(x_i, y_i^+, y_i^-)\}_{i=1}^N$, SFT policy π_0, reference policy π_{ref}
Ensure: Trained policy π_θ

Attach a frozen-base PiSSA probe with parameters ϕ to π_0. ▷ Defines the gradient space
Gradient-based weight precomputation
$g^+_{\text{pair},i} \leftarrow G^+(x_i, y_i^+)$ for $i = 1, \ldots, N$ ▷ Preferred direction of each pair
$g^+_{\text{global}} \leftarrow \frac{1}{N} \sum_{i=1}^N g^+_{\text{pair},i}$ ▷ Dataset-level preferred direction
for $i = 1, \ldots, N$ **do**
  for $t = 1, \ldots, |y_i^-|$ **do** ▷ Alignment scores for each token t
    $g_t^- \leftarrow \nabla_\phi \log \pi_\phi(y^-_{i,t} \mid x_i, y^-_{i,<t})$
    $s_t \leftarrow \max\left(\langle g_t^-, g^+_{\text{pair},i}\rangle, \langle g_t^-, g^+_{\text{global}}\rangle\right)$ ▷ Protect local or global preferred behavior
  end for
  $m \leftarrow \max_t |s_t|$ ▷ Largest absolute score of this response
  for $t = 1, \ldots, |y_i^-|$ **do** ▷ Token weights
    $w_{i,t} \leftarrow \sigma(-s_t/m)$ ▷ Higher alignment ⇒ weaker penalty
  end for
end for
Weighted DPO training
Initialize π_θ.
while training **do**
  Sample i; compute the chosen log-ratio r^+ with uniform token weights.
  $r^- \leftarrow \sum_t w_{i,t} \left[\log \pi_\theta(y^-_{i,t} \mid x_i, y^-_{i,<t}) - \log \pi_{\text{ref}}(y^-_{i,t} \mid x_i, y^-_{i,<t})\right]$
  Update θ using $\mathcal{L} = -\log \sigma\left(\beta(r^+ - r^-)\right)$.
end while
return π_θ

4 Experimental Setup

4.1 Experimental Details

We use *Olmo-3-7B-Instruct-SFT*[1], the supervised fine-tuning (SFT) checkpoint of the 7B OLMo 3 Instruct model (Olmo et al., 2025), as the initialization for all experiments. OLMo 3 is a fully open model family, with publicly available model weights, intermediate checkpoints, and training data, making it suitable for controlled studies of optimization dynamics.

For preference training, we use a filtered subset of *Dolci-Instruct-DPO*[2], the preference-training dataset used for the OLMo 3 Instruct DPO stage (Olmo et al., 2025). For a controlled setting, we retain only English, single-turn user-assistant preference pairs, and restrict the total textual content to at most 2048 tokens. The resulting dataset contains approximately 209K preference pairs (80% of the original dataset). We use the standard OLMo 3 chat template and mask prompt tokens when computing completion losses.

For our method, we perform full fine-tuning for one epoch with a maximum sequence length of 2048, learning rate 1×10^{-5}, effective batch size 128, $\beta = 0.1$, and warmup ratio 0.1. We use a linear learning-rate schedule, 8-bit AdamW, `bfloat16`. Unless stated otherwise for a specific ablation or experiment, this constitutes our default experimental setup. For gradient-weight computation, we attach a rank-64 PiSSA adapter with $\alpha = 128$, zero dropout, and all linear layers as target modules. The adapter is used exclusively as a gradient probe; preference optimization itself remains a full-model fine-tune.

Precomputing the alignment scores over all the preference pairs costs approximately 63 GPU-hours on H200 GPUs: one reverse-mode pass per example to obtain the chosen direction, followed by two forward-mode (JVP) passes for the paired and dataset-level projections (see Appendix A.4).

[1] `https://huggingface.co/allenai/Olmo-3-7B-Instruct-SFT`
[2] `https://huggingface.co/datasets/allenai/Dolci-Instruct-DPO`

The precomputation is performed once per dataset and is reused across every training run in our experiments. For comparison, a single full-model training run takes approximately 40 GPU-hours.

4.2 Baselines

We compare against the starting SFT checkpoint, standard DPO (Rafailov et al., 2023), 3 sequence-level variants (i.e., IPO (Azar et al., 2024), KTO (Ethayarajh et al., 2024) and SimPO (Meng et al., 2024b)), and 3 token-weighting methods (i.e., TIS-DPO (Liu et al., 2025), TI-DPO (Yang et al., 2026) and AAO (Li et al., 2025)). All methods start from the same SFT checkpoint and are trained for one epoch on the same preference data. Method-specific hyperparameters follow the original papers or their official implementations. Appendix C lists all values and the few deviations.

4.3 Evaluation Benchmarks

We evaluate all models using a subset of the OLMES benchmark suite (Gu et al., 2025), following the same evaluation framework used in the OLMo 3 paper (Olmo et al., 2025). All results are reported across 3 runs with different seeds. We use the official OLMES evaluation recipes for OLMo 3 and select benchmarks spanning four broad capability groups: **Mathematics:** AIME 2024 (Zhang & Math-AI, 2024) and 2025 (Zhang & Math-AI, 2025), MATH (Hendrycks et al., 2021b), and Omega 500 (Sun et al., 2026); **Reasoning:** AGI Eval English (Zhong et al., 2024) and BigBenchHard-BBH (Suzgun et al., 2023) with chain-of-thought prompting; **Code:** HumanEval+ (Chen et al., 2021; Liu et al., 2023), LiveCodeBench (Jain et al., 2025), and MBPP+ (Austin et al., 2021; Liu et al., 2023); **Question Answering:** GPQA (Rein et al., 2024) and MMLU (Hendrycks et al., 2021a) with chain-of-thought prompting.

5 Results

5.1 Comparison with Existing Methods

Table 1 compares GAW-PO against standard DPO and existing preference-optimization and token-weighting approaches using the experimental setup from Section 4. Our method achieves the strongest overall performance, with an average score of **41.99**. This corresponds to an absolute improvement of **0.97 points** over standard DPO (41.02) and **0.65 points** over TIS-DPO (41.34), the strongest competing method on average. Moreover, the gains are distributed across several capability groups; GAW-PO obtains the best result on 8 of the 11 evaluated benchmarks.

The largest improvements are observed on reasoning and coding tasks. On BBH, our method reaches **49.81**, improving over standard DPO by 4.36 points and over the strongest competing baseline by 4.31 points. Similarly, on LiveCodeBench, we obtain **24.78**, compared with 21.36 for DPO and 21.88 for TIS-DPO. We also observe consistent gains on HumanEval+ (73.52), MATH (70.35), AGI Eval (61.15), MBPP+ (57.73), and MMLU (67.92). These results indicate that reducing harmful rejected-token updates can improve performance across different downstream tasks. Appendix D.1 shows that the weights fall on the tokens where a rejected response diverges from the chosen one, sparing correct content.

The improvement is not universal across all benchmarks. TIS-DPO performs best on AIME 2025 and GPQA, while IPO obtains the highest score on Omega 500. In particular, our method underperforms on GPQA and Omega 500. Nevertheless, the stronger performance across the majority of benchmarks results in the highest aggregate score among all evaluated methods.

Overall, these results support our hypothesis that token weights should depend on their *interaction with preferred update directions*, rather than only on token-level importance in isolation. By downweighting rejected tokens whose gradients are compatible with preferred behavior and concentrating the negative signal on conflicting updates, our method produces a more targeted preference-optimization signal and improves overall downstream performance.

Table 1: Performance comparison across evaluation benchmarks: mean performance (%) across 3 runs, standard deviation shown in gray. Best results per benchmark are **bolded**.

Benchmark	SFT	DPO	IPO	KTO	SimPO	TIS-DPO	TI-DPO	AAO	GAW-PO (Ours)
AIME 2024	5.73 ± 0.55	6.28 ± 0.80	6.25 ± 0.21	5.66 ± 0.97	6.63 ± 0.33	6.46 ± 0.81	6.53 ± 0.51	6.67 ± 0.38	**7.36** ± 0.52
AIME 2025	6.88 ± 1.04	8.33 ± 1.00	7.57 ± 0.66	7.81 ± 0.65	7.95 ± 0.69	**8.47** ± 1.16	8.44 ± 0.99	7.88 ± 0.99	7.95 ± 0.42
MATH	67.70 ± 0.27	68.69 ± 0.38	68.07 ± 0.73	67.77 ± 0.47	68.61 ± 0.81	69.41 ± 0.36	69.30 ± 0.47	68.59 ± 0.37	**70.35** ± 0.24
Omega 500	11.53 ± 0.76	12.27 ± 1.03	**12.47** ± 0.95	11.87 ± 1.85	11.93 ± 0.95	12.33 ± 0.92	12.13 ± 0.76	11.93 ± 0.99	11.60 ± 1.31
AGI Eval	58.51 ± 0.58	60.31 ± 1.08	59.45 ± 0.62	59.46 ± 0.59	58.47 ± 1.40	60.12 ± 0.62	59.75 ± 0.34	59.14 ± 0.29	**61.15** ± 0.74
BBH (CoT)	44.40 ± 0.77	45.45 ± 0.55	45.37 ± 0.85	45.12 ± 0.09	45.50 ± 0.64	45.33 ± 0.11	45.13 ± 0.82	44.87 ± 0.69	**49.81** ± 0.21
HumanEval+	68.84 ± 0.48	71.63 ± 0.94	71.12 ± 0.58	69.51 ± 0.44	70.98 ± 0.79	71.87 ± 0.27	71.59 ± 1.06	71.32 ± 0.90	**73.52** ± 0.41
LiveCodeBench	20.30 ± 0.34	21.36 ± 0.26	20.72 ± 0.10	20.72 ± 0.14	21.13 ± 0.13	21.88 ± 0.25	21.24 ± 0.21	20.92 ± 0.35	**24.78** ± 0.37
MBPP+	56.38 ± 0.37	56.71 ± 0.31	56.78 ± 0.33	56.49 ± 0.12	56.67 ± 0.27	57.13 ± 0.68	56.71 ± 0.19	56.45 ± 0.31	**57.73** ± 1.17
GPQA	31.47 ± 1.74	33.26 ± 1.36	31.99 ± 2.68	31.92 ± 1.61	32.14 ± 0.67	**34.08** ± 2.41	31.55 ± 2.41	30.88 ± 0.52	29.76 ± 0.26
MMLU (CoT)	66.75 ± 0.32	66.91 ± 0.49	67.48 ± 0.17	66.86 ± 0.11	66.92 ± 0.59	67.70 ± 0.19	67.63 ± 0.28	67.09 ± 0.19	**67.92** ± 0.25
Average	39.86	41.02	40.66	40.29	40.63	41.34	40.91	40.52	**41.99**

5.2 Robustness to the DPO Regularization Strength

The DPO parameter β (see Equation 1) controls the strength of the implicit regularization toward the reference policy. Larger values constrain the optimized policy more strongly, whereas smaller values permit larger deviations from the reference model. In practice, DPO is commonly trained with moderate values of $\beta \approx 0.1$, since aggressive preference optimization can lead to substantial degradation due to policy collapse. Under our motivating hypothesis, part of this degradation may arise because stronger optimization also amplifies the penalty applied to locally correct tokens in rejected responses, suppressing both useful behavior and genuinely undesirable parts of the response.

We investigate whether our token-level weighting improves robustness in this regime by sweeping $\beta \in \{0.2, 0.1, 0.05, 0.02, 0.01, 0.005, 0.002\}$, using the same learning rate of 1×10^{-5} for both standard DPO and GAW-PO. Figure 2 reports the average score across all 11 evaluation benchmarks.

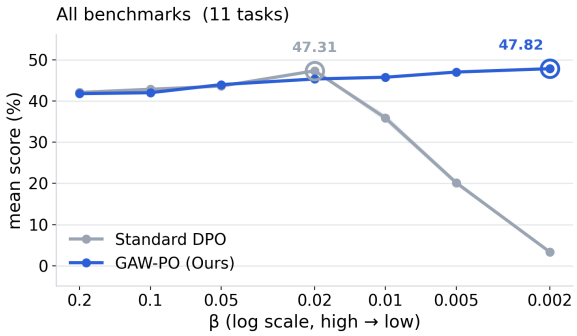

Figure 2: Average benchmark performance for different values of the DPO regularization parameter β. Standard DPO exhibits a narrow optimum around $\beta = 0.02$ and rapidly degrades under more aggressive optimization, whereas our method remains stable and continues to improve as β decreases.

The two methods behave similarly in the strongly regularized regime. Standard DPO reaches its best performance around $\beta = 0.02$, but deteriorates sharply once β is decreased further. In contrast, our method remains stable over the entire sweep and improves as the regularization is relaxed. The difference becomes pronounced at $\beta = 0.01$ and grows rapidly for $\beta = 0.005$ and $\beta = 0.002$. At the most aggressive setting, standard DPO loses almost all of its performance, whereas GAW-PO achieves its best aggregate result. Notably, GAW-PO at $\beta = 0.002$ also surpasses the best standard-DPO configuration observed anywhere in the sweep. The category-level results detailed in Appendix E.1 reveal the same pattern across mathematics, reasoning, coding, and QA.

These results indicate that our weighting scheme substantially enlarges the range of β values over which preference optimization remains effective. Standard DPO requires relatively strong regularization to prevent destructive updates, resulting in a narrow range of effective optimization strengths. By reducing the contribution of rejected tokens whose gradients are aligned with preferred behavior, GAW-PO permits substantially larger deviations from the reference policy without loss in down-

Table 2: Average performance for paired+global and clustered preferred directions at $\beta = 0.002$. We report mean performance across 3 runs. Full per-benchmark results are in Appendix E.2.

Metric	Mean	K=2	K=4	K=8
Average	47.82	46.60	**47.94**	47.89

stream capability. This is consistent with our hypothesis that part of the instability of aggressive DPO originates from negative updates applied to behavior that should be preserved. Random or uniform weights with the same mean lose relative to gradient-aligned weights, so the gain is not due to reduced penalty magnitude alone (see Appendix D.2).

5.3 Multiple Preferred Update Directions

Our main method compares each rejected-token gradient against both its paired chosen gradient and a single aggregate gradient over all chosen responses. However, a single dataset-level mean may be too coarse for a heterogeneous preference dataset, where different types of preferred behavior can induce distinct update directions. We therefore investigate a clustered variant in which chosen gradients are partitioned into K groups in the parameter space. For each cluster, we compute a centroid representing its average preferred update. The score of a rejected token becomes $s_t^{\text{cluster}} = \max\left(\left\langle g_t^-, g_{\text{pair}}^+ \right\rangle, \max_{k=1,\dots,K} \left\langle g_t^-, c_k^+ \right\rangle\right)$, where c_k^+ denotes the chosen-gradient centroid of cluster k. Thus, a rejected token is protected if it aligns either with its paired chosen response or with any recurring preferred direction in the dataset.

Among the clustered variants, $K = 4$ performs best, reaching **47.94** compared with 47.82 for the original paired+global formulation. $K = 8$ also slightly improves over the global mean, while $K = 2$ performs worse. The gains for $K = 4$ are most visible in question answering and mathematics, suggesting that multiple preferred directions can be beneficial, although performance does not improve monotonically with K. Additional clustering details and full per-benchmark results are provided in Appendix B and E.2.

6 Conclusion and Future Work

We introduced a gradient-aligned token reweighting approach for DPO that accounts for the effect of penalizing individual rejected tokens on preferred behavior. Rather than estimating token importance in isolation, GAW-PO compares rejected-token gradients with preferred update directions and reduces the penalty on tokens whose suppression would interfere with learning the chosen response. We make this computation practical through a low-rank PiSSA gradient probe, while retaining full-model preference optimization during training.

Across our evaluation suite, the proposed method achieves the highest average performance among the considered preference-optimization methods, outperforming standard DPO and the strongest competing baseline. More importantly, the method remains effective under substantially more aggressive preference optimization: as β decreases, standard DPO degrades sharply, whereas GAW-PO remained stable and often improved. We additionally explored representing preferred behavior through multiple clustered update directions. While this provides modest gains for some values of K, the results also showed that increasing the number of preferred directions is not sufficient by itself; the structure and quality of these directions are also important.

In terms of future work, we plan to extend our method beyond pairwise objectives such as DPO to online reinforcement-learning methods, particularly Group Relative Policy Optimization (GRPO) (Shao et al., 2024). Although GRPO assigns updates based on relative rewards across multiple sampled responses, low-reward trajectories may still contain useful tokens or reasoning steps that are also supported by higher-reward responses. Our gradient-alignment framework could provide token-level credit assignment by downweighting negative token updates whose gradients align with high-advantage responses, while preserving stronger penalties for conflicting tokens.

Finally, Appendix F discusses the study's limitations, from the cost of the offline weight computation to the scope of our evaluation, each pointing to a direction for future work.

AI USE DISCLOSURE

LLMs were used as assistive tools in the preparation of this manuscript and the accompanying code. Specifically, we used ChatGPT 5.0+ (OpenAI, [3]) and Claude Opus 5.0+ (Anthropic, [4]) to assist with drafting and revising portions of the manuscript. Their use included grammar and spell checking, improving clarity and phrasing, reducing repetition, and general language polishing. Claude was additionally used to assist with suggesting and debugging portions of the experimental code, as well as with the generation and refinement of the teaser figure used in the manuscript. LLMs were used only as assistive tools; all generated text and code suggestions were reviewed, verified, and, where necessary, modified by the authors. The authors remain fully responsible for the content of the manuscript, the implementation, the experimental methodology, and the reported results.

REFERENCES

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

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. A general theoretical paradigm to understand learning from human preferences. In *International conference on artificial intelligence and statistics*, pp. 4447–4455. PMLR, 2024.

Moses Charikar, Kevin Chen, and Martin Farach-Colton. Finding frequent items in data streams. In *International colloquium on automata, languages, and programming*, pp. 693–703. Springer, 2002.

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. Evaluating large language models trained on code. 2021.

Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Wenlong Deng, Yi Ren, Muchen Li, Danica J Sutherland, Xiaoxiao Li, Christos Thrampoulidis, et al. On the effect of negative gradient in group relative deep reinforcement optimization. *Advances in Neural Information Processing Systems*, 38:73828–73856, 2026.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. Model alignment as prospect theoretic optimization. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 12634–12651, 2024.

Yuling Gu, Oyvind Tafjord, Bailey Kuehl, Dany Haddad, Jesse Dodge, and Hannaneh Hajishirzi. OLMES: A standard for language model evaluations. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 5020–5048, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-195-7. doi: 10.18653/v1/2025.findings-naacl.282. URL https://aclanthology.org/2025.findings-naacl.282/.

[3] https://chatgpt.com/

[4] https://claude.ai/

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

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

Naman Jain, 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. In *International Conference on Learning Representations*, volume 2025, pp. 58791–58831, 2025.

Jaehun Jung, Seungju Han, Ximing Lu, Skyler Hallinan, David Acuna, Shrimai Prabhumoye, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, and Yejin Choi. Prismatic synthesis: Gradient-based data diversification boosts generalization in llm reasoning. *Advances in Neural Information Processing Systems*, 38:90649–90685, 2026.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pp. 1885–1894. Pmlr, 2017.

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *arXiv preprint arXiv:2406.18629*, 2024.

Jian Li, Shenglin Yin, Yujia Zhang, Alan Zhao, Xi Chen, Xiaohui Zhou, and Pengfei Xu. Ambiguity awareness optimization: towards semantic disambiguation for direct preference optimization. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 9064–9074, 2025.

Aiwei Liu, Haoping Bai, Zhiyun Lu, Yanchao Sun, Xiang Kong, Xiaoming Wang, Jiulong Shan, Albin Madappally Jose, Xiaojiang Liu, Lijie Wen, et al. Tis-dpo: Token-level importance sampling for direct preference optimization with estimated weights. In *International Conference on Learning Representations*, volume 2025, pp. 51339–51368, 2025.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. volume 36, pp. 21558–21572. 2023.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. Pissa: Principal singular values and singular vectors adaptation of large language models. *Advances in Neural Information Processing Systems*, 37:121038–121072, 2024a.

Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235, 2024b.

Team Olmo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, Jacob Morrison, Jake Poznanski, Kyle Lo, Luca Soldaini, Matt Jordan, Mayee Chen, Michael Noukhovitch, Nathan Lambert, Pete Walsh, Pradeep Dasigi, Robert Berry, Saumya Malik, Saurabh Shah, Scott Geng, Shane Arora, Shashank Gupta, Taira Anderson, Teng Xiao, Tyler Murray, Tyler Romero, Victoria Graf, Akari Asai, Akshita Bhagia, Alexander Wettig, Alisa Liu, Aman Rangapur, Chloe Anastasiades, Costa Huang, Dustin Schwenk, Harsh Trivedi, Ian Magnusson, Jaron Lochner, Jiacheng Liu, Lester James V. Miranda, Maarten Sap, Malia Morgan, Michael Schmitz, Michal Guerquin, Michael Wilson, Regan Huff, Ronan Le Bras, Rui Xin, Rulin Shao, Sam Skjonsberg, Shannon Zejiang Shen, Shuyue Stella Li, Tucker Wilde, Valentina Pyatkin, Will Merrill, Yapei Chang, Yuling Gu, Zhiyuan Zeng, Ashish Sabharwal, Luke Zettlemoyer, Pang Wei Koh, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. Olmo 3, 2025. URL https://arxiv.org/abs/2512.13961.

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. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744, 2022.

Arka Pal, Deep Karkhanis, Samuel Dooley, Manley Roberts, Siddartha Naidu, and Colin White. Smaug: Fixing failure modes of preference optimisation with dpo-positive. *arXiv preprint arXiv:2402.13228*, 2024.

Richard Y Pang, Weizhe Yuan, Kyunghyun Cho, He He, Sainbayar Sukhbaatar, and Jason Weston. Iterative reasoning preference optimization. *Advances in Neural Information Processing Systems*, 37:116617–116637, 2024.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. *Advances in neural information processing systems*, 33: 19920–19930, 2020.

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. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Noam Razin, Sadhika Malladi, Adithya Bhaskar, Danqi Chen, Sanjeev Arora, and Boris Hanin. Unintentional unalignment: Likelihood displacement in direct preference optimization. In *International Conference on Learning Representations*, volume 2025, pp. 24791–24834, 2025.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Yiyou Sun, Shawn Hu, Georgia Zhou, Ken Zheng, Hanna Hajishirzi, Nouha Dziri, and Dawn Song. Omega: Can llms reason outside the box in math? evaluating exploratory, compositional, and transformative generalization. *Advances in Neural Information Processing Systems*, 38, 2026.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed H Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051, 2023.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. LESS: Selecting influential data for targeted instruction tuning. In *International Conference on Machine Learning (ICML)*, 2024.

Ning Yang, Hai Lin, Yibo Liu, Baoliang Tian, Guoqing Liu, and Haijun Zhang. Token-importance guided direct preference optimization. In *International Conference on Learning Representations*, volume 2026, pp. 13888–13910, 2026.

Yongcheng Zeng, Guoqing Liu, Weiyu Ma, Ning Yang, Haifeng Zhang, and Jun Wang. Token-level direct preference optimization. *arXiv preprint arXiv:2404.11999*, 2024.

Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2024, 2024.

Yifan Zhang and Team Math-AI. American invitational mathematics examination (aime) 2025, 2025.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied, Weizhu Chen, and Nan Duan. AGIEval: A human-centric benchmark for evaluating foundation models. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 2299–2314, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-naacl.149. URL https://aclanthology.org/2024.findings-naacl.149/.

A Additional Gradient-Weighting Details

A.1 Relation to the DPO derivation.

Standard DPO is derived as an exact reparameterization of the KL-constrained sequence-level bandit objective, where the implicit reward $r_\theta(x, y) = \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}$ is given by the log-ratio of two normalized distributions (Rafailov et al., 2023). The weighted rejected term $r_w(x, y^-) = \beta \sum_t w_t \big[\log \pi_\theta(y_t^- \mid x, y_{<t}^-) - \log \pi_{\text{ref}}(y_t^- \mid x, y_{<t}^-)\big]$ does not have this form. Therefore, the weighted learning objective should be viewed as a contrastive surrogate rather than as the exact dual of a KL-constrained sequence-level objective.

However, its impact on optimization is straightforward since the weights are detached constants,

$$\nabla_\theta \mathcal{L} = -\beta\, \mathbb{E}\Big[\sigma\big(-\beta(r^+ - r_w^-)\big)\Big(\nabla_\theta \log \pi_\theta(y^+ \mid x) - \sum_t w_t\, \nabla_\theta \log \pi_\theta(y_t^- \mid x, y_{<t}^-)\Big)\Big], \quad (9)$$

is simply the standard DPO gradient with the gradient contribution of each rejected token scaled by w_t. The method can therefore be understood as applying token-level credit assignment to the DPO update.

Consequently, w_t can be viewed as inducing a token-dependent regularization coefficient βw_t for the rejected response. This interpretation is not an exact derivation, because the coefficient is not defined consistently as a function of state across both chosen and rejected trajectories. Nevertheless, it places our objective in the same general family as token-level variants such as TDPO (Zeng et al., 2024) and TIS-DPO (Liu et al., 2025), which likewise depart from the sequence-level dual in order to obtain finer-grained credit assignment.

A.2 PiSSA Gradient Probe

Computing a separate gradient for every rejected token over all parameters of a 7B model is prohibitively expensive. We therefore measure gradient alignment in a low-rank parameter space defined by a PiSSA-initialized adapter (Meng et al., 2024a) attached to the frozen SFT model. Importantly, this adapter is used only as a *gradient probe*: it is never optimized, and the subsequent preference-optimization stage remains a full-model fine-tune.

PiSSA initializes the two low-rank factors from a truncated singular-value decomposition of each target weight matrix. For

$$W = U \Sigma V^\top, \quad (10)$$

the rank-r component is represented using

$$A = \sqrt{\Sigma_r} V_r^\top, \qquad B = U_r \sqrt{\Sigma_r}, \quad (11)$$

while the corresponding contribution is subtracted from the frozen base weight. Consequently, the function represented by the model is unchanged at initialization, while both low-rank factors contain non-zero values and admit informative gradients.

In all experiments, we use a rank-64 PiSSA adapter with scaling parameter $\alpha = 128$, zero dropout, and all linear layers as target modules. The same frozen probe is used to compute the chosen-response directions and rejected-token gradients.

A.3 Construction of the Chosen Direction

For a chosen token y_i^+ with logits z_i and target probability $p_i^+ = \mathrm{softmax}(z_i)_{y_i^+}$, the gradient of the token log-likelihood with respect to the logits is

$$\frac{\partial \log p_i^+}{\partial z_i} = \mathrm{onehot}(y_i^+) - \mathrm{softmax}(z_i), \tag{12}$$

whose L_1 norm is $(1 - p_i^+) + \sum_{j \neq y_i^+} p_{i,j} = 2(1 - p_i^+)$. The parameter gradient is $g_i^+ = (\partial z_i / \partial \phi)^\top \, \partial \log p_i^+ / \partial z_i$. The weight $\alpha_i = 1/(2(1 - p_i^+))$ in Equation 7 therefore rescales each token so that its logit-level error vector has unit L_1 norm. It does not normalize the parameter gradient itself.

Without the rescaling, $\|\partial \log p_i^+ / \partial z_i\|_1 = 2(1 - p_i^+)$ vanishes for tokens the model already predicts with high confidence, and the sum $\sum_i g_i^+$ is dominated by the small number of tokens with low p_i^+. Since g^+ serves as the reference against which every rejected token is scored, this would make the alignment scores depend on the most surprising chosen tokens rather than on the chosen response as a whole. The rescaled sum instead approximates an unweighted average over token gradient directions.

A.4 Efficient Alignment Projection Computation

Explicitly computing and storing a gradient for every rejected token would require a matrix with dimensions $T^- \times D$, where $D \approx 1.6 \times 10^8$ in our PiSSA probe. However, our scoring function never requires the full rejected-token gradients themselves; it only requires their inner products with a small number of preferred directions.

We therefore compute these quantities directly using forward-mode automatic differentiation. Let

$$f(\phi) = \left[\log \pi_\phi(y_1^- \mid x), \ldots, \log \pi_\phi(y_{T^-}^- \mid x, y_{<T^-}^-)\right] \tag{13}$$

denote the vector of rejected-token log-likelihoods. For any fixed preferred direction v, a Jacobian vector product gives

$$J_f(\phi) v = \left[\left\langle g_1^-, v \right\rangle, \ldots, \left\langle g_{T^-}^-, v \right\rangle\right], \tag{14}$$

where $g_t^- = \nabla_\phi \log \pi_\phi(y_t^- \mid x, y_{<t}^-)$. Thus, a single directional pass computes the alignment scores for all rejected tokens simultaneously.

For the main method, we evaluate this projection for the pair-specific chosen direction g_{pair}^+ and the dataset-level direction g_{global}^+, and then take their element-wise maximum. Consequently, we never materialize the full $T^- \times D$ rejected-gradient matrix; only the resulting scalar alignment scores are stored. The number of differentiation passes therefore depends on the number of preferred reference directions rather than on the number of rejected tokens.

B Multiple Preferred Update Directions

B.1 Clustering Procedure

The main method represents dataset-level preferred behavior using a single mean chosen gradient. To study whether the chosen-gradient space contains additional structure, we construct multiple preferred reference directions through clustering.

Computing and storing the full D-dimensional chosen gradient for every preference example would be impractical. We therefore first project each chosen-response gradient into a 1000-dimensional CountSketch representation (Charikar et al., 2002). These sketches are used only for clustering.

We apply spherical k-means to the normalized sketch vectors. Spherical clustering groups examples according to gradient direction rather than Euclidean distance or raw gradient magnitude. Given

assignments $\mathcal{C}_1, \ldots, \mathcal{C}_K$, we subsequently recompute each centroid in the original PiSSA parameter space,

$$c_k^+ = \frac{1}{|\mathcal{C}_k|} \sum_{e \in \mathcal{C}_k} g_e^+. \tag{15}$$

The full-space recomputation is necessary because sketch-space centroids cannot be directly compared with token gradients in the PiSSA parameter space.

For a rejected token, the clustered score is

$$s_t^{\text{cluster}} = \max\left(\left\langle g_t^-, g_{\text{pair}}^+ \right\rangle, \max_{k=1,\ldots,K} \left\langle g_t^-, c_k^+ \right\rangle \right). \tag{16}$$

Thus, each rejected token independently selects the preferred direction with which it has the largest alignment. We do not first assign the rejected example to a particular chosen-gradient cluster.

We evaluate $K \in \{2, 4, 8\}$. All clustered variants are trained at $\beta = 0.002$, the most aggressive setting of the sweep in Section 5.2 and the one at which our method attains its best aggregate result; all other hyperparameters follow the default setup of Section 4.1.

B.2 Efficient Cluster Projections

The clustered variant uses the same projection mechanism as the main method. The only difference is that the rejected-token gradients are compared against multiple dataset-level preferred directions rather than a single global mean.

For each cluster centroid c_k^+, we compute

$$J_f(\phi) c_k^+ = \left[\left\langle g_1^-, c_k^+ \right\rangle, \ldots, \left\langle g_{T^-}^-, c_k^+ \right\rangle \right], \tag{17}$$

using one Jacobian vector product. Repeating this operation for the K cluster centroids yields all token-centroid alignment scores without explicitly constructing any rejected-token gradient vectors. The pair-specific projection is computed in the same way, after which the final score is obtained by taking the element-wise maximum over the paired direction and all cluster directions,

$$s_t^{\text{cluster}} = \max\left(\left\langle g_t^-, g_{\text{pair}}^+ \right\rangle, \max_{k=1,\ldots,K} \left\langle g_t^-, c_k^+ \right\rangle \right). \tag{18}$$

Thus, the clustered extension increases the number of directional passes from two in the main method to $K + 1$, while retaining the same memory-efficient computation and avoiding per-token reverse-mode gradient materialization.

C Baseline Hyperparameters

Following the OLMo 3 7B Instruct DPO recipe (Olmo et al., 2025), all baselines are trained for one epoch with an effective batch size of 128 and a linear learning-rate schedule with 10% warmup. All baselines use 8-bit AdamW and, unless stated otherwise below, a learning rate of 2×10^{-6}.

DPO and IPO. Both use $\beta = 0.1$. The IPO paper does not give a recipe for LLM training, so we use $\tau = 0.1$, the same value as β for DPO.

KTO. Each preference pair is split into one desirable and one undesirable example, which gives a balanced dataset, so we set $\lambda_D = \lambda_U = 1$ and $\beta = 0.1$. We use a learning rate of 1×10^{-6}, the maximum recommended for $\beta = 0.1$ in the TRL implementation of KTO, since higher values degrade training.

SimPO. We use $\beta = 10$ and $\gamma/\beta = 0.3$ from the official implementation. With its learning rate of 1×10^{-6} the model did not learn in our setup, so we use the shared 2×10^{-6}.

TIS-DPO. We use the variant in which token weights are estimated from two contrastive models obtained with DPO and reverse DPO, with $\mu = \pm 1$ and the log-ratio clamped to $[-0.5, 1.5]$ as in the original paper, and the TDPO$_2$ objective with $\alpha = 0.5$. Weights are normalized per sequence.

TI-DPO. Token weights mix gradient attribution with a Gaussian position prior, with $\lambda = 0.7$ and $\sigma = T/4$, normalized to mean one per sequence. We use $\beta = 0.1$ and the TDPO$_2$ objective with $\alpha = 0.5$. Our runs match the *no triplet loss* framework of the original paper. We use the learning rate of 5×10^{-6} from the authors' configuration.

AAO. We use the smoothing parameter $\alpha = 200$ from the original paper. The paper learns the ambiguity thresholds with a small network; we fix them to $a = 0.6$ and $b = 0.3$.

D Qualitative and Control Analyses of Token Weights

D.1 Qualitative Examples

Table 3 shows per-token weights for three rejected completions covering instruction violation, fabricated content, and factual error. Several patterns emerge.

First, larger weights tend to appear near the point where the rejected response diverges from the preferred behavior, rather than on tokens that follow from that error. In (a), the opening fence receives a larger weight, while the closing fence will not be penalized as harshly. Similarly, `ASS` and `NEG`, which introduce and label a fabricated combination, receive high weights, whereas the tokens completing `ASS1` and `siRNA` receive low weights, as they are a natural completion after the starting tokens.

Second, correctly reproduced content generally receives small weights. The copied drug names in (a), the JSON structure in (a) and (b), and the formulas in (c) all receive low weights. Under standard DPO, these tokens would receive the same rejected-response penalty as the erroneous tokens.

Third, the weights depend on the token's role relative to the chosen response rather than on its surface form alone. In (c), `acid` receives a low weight where it is correct and high where it is incorrect. The same fence token receives a large weight in (a), where it violates an explicit instruction, and low in (b), where no such instruction is present.

D.2 Uniform and Random Weight Controls

Because $w_t \leq 1$, our weighted objective reduces the total negative contribution of rejected tokens relative to standard DPO. To separate the effect of this overall attenuation from the effect of assigning different weights to different tokens, we train two controls at $\beta = 0.002$ using the setup of Section 4.1 (the same setup as our best model variant). **Uniform** assigns $w_t = 0.489$, the mean of our learned weights, to every rejected token. **Random** samples $w_t \sim \mathcal{U}[0, 1]$ independently for each token once before training, matching the expected attenuation while removing any relationship between the weight and the token. Table 4 reports the results.

Uniform weighting recovers much of the stability, suggesting that reducing the magnitude of the negative update is itself beneficial. However, attenuation alone does not explain the full improvement: gradient-aligned weights outperform uniform weights by 2.09 points and random weights by 3.36 points.

The largest differences appear on tasks where rejected responses can contain substantial correct structure shared with the chosen response. Relative to GAW-PO, uniform and random weighting are lower by 12.95/21.47 points on HumanEval+, 2.83/3.83 points on BBH, and 1.95/2.12 points on MMLU, while the mathematics benchmarks are generally within one standard deviation across variants. Random weighting also performs below uniform weighting overall. These controls suggest that performance at small β certainly depends on assigning the penalty selectively across tokens.

Table 3: Per-token weights assigned by our method on three rejected completions. Shading is monotone in the weight a token receives (pale = low, deep = high). The weight concentrates on the tokens listed under **Errors**, while correctly transcribed content receives near-uniform low weight.
*The prompts are summarized and example (c) is truncated for readability.

(a) Biomedical relation extraction

Prompt Extract every drug combination in the sentence *"Reduction of **ASS1** expression by **siRNA** significantly sensitized mesothelioma spheroids to the pro-apoptotic effects of **bortezomib** and of **cisplatin** plus **pemetrexed**"*, copying drug names verbatim and labelling each POS, NEG or COMB. *"In your output, return only the json array and no other text."*

Chosen `[["bortezomib", "cisplatin", "pemetrexed", "COMB"], ["cisplatin", "pemetrexed", "COMB"]]`

Rejected
```
```json
[
  ["ASS1", "siRNA", "NEG"],
  ["bortezomib", "cisplatin", "pemetrexed", "POS"]
]
```<|endoftext|>
```

Errors The ``` fence breaks the instruction to return only the array; `ASS1` is a gene and `siRNA` a technique, so the first combination is fabricated; and the genuine combination is labelled POS rather than COMB.

(b) Structured numerical answer

Prompt Flights of 8, 4 and 6 hours with a 2-hour delay at each border. Report the total travel time as JSON.

Chosen
```
```json
{"total_travel_time_hours": 24}
```
```

Rejected
```
```json
{
  "total_time": "18 hours",
  "border_delays": 2
}
```<|endoftext|>
```

Errors The total is returned as a string rather than a number, `18` counts only the flights and drops the delays, and `border_delays` is a field nobody asked for.

(c) Chemical classification

Prompt Classify each oxide as strongly acidic, weakly acidic, neutral, amphoteric, weakly basic or strongly basic: … (j) OsO_4 (k) Al_2O_3 …

Chosen **(j)** OsO_4 – strongly *acidic*; forms osmic acid in water. **(k)** Al_2O_3 – *amphoteric*; reacts with both acid and base.

Rejected `[...] (j) OsO_4 (OsO): Strongly basic (acidic) (k) Al_2O_3 (Aluminum oxide): Strongly basic (acidic) [...]`

Errors Both oxides are called *strongly basic*: OsO_4 is acidic and Al_2O_3 amphoteric. Each line then contradicts its own label in the following parenthesis.

E Additional Experiments - Detailed Results

E.1 Category Breakdown of Robustness to the DPO Regularization Strength

This section provides a more detailed view of the β sweep presented in Section 5.2. Figure 3 breaks down the aggregate results by benchmark category, showing how standard DPO and GAW-PO behave across mathematics, reasoning, coding, and question answering as the optimization becomes more aggressive.

Table 4: Comparison of gradient-based, uniform, and random rejected-token weights. We report mean performance (%) across 3 runs, with standard deviation shown in gray. Best results per benchmark are **bolded**.

Benchmark	GAW-PO	Uniform Weights	Random Weights
AIME 2024	16.77 ± 0.79	**17.12** ± 0.79	16.74 ± 0.52
AIME 2025	**17.53** ± 0.68	16.91 ± 0.37	17.01 ± 0.81
MATH	76.63 ± 0.39	**76.71** ± 0.35	76.61 ± 0.36
Omega 500	12.33 ± 0.70	13.20 ± 0.92	**13.53** ± 0.81
AGI Eval	**65.49** ± 0.24	65.12 ± 0.41	64.26 ± 0.67
BBH (CoT)	**67.79** ± 0.49	64.96 ± 0.40	63.96 ± 0.14
HumanEval+	**75.35** ± 0.57	62.40 ± 0.88	53.88 ± 0.36
LiveCodeBench	**30.01** ± 0.79	28.01 ± 0.19	26.15 ± 0.25
MBPP+	**59.11** ± 0.29	55.85 ± 0.56	54.20 ± 0.21
GPQA	**35.04** ± 0.39	34.75 ± 1.52	34.82 ± 1.24
MMLU (CoT)	**69.98** ± 0.57	68.03 ± 0.35	67.86 ± 0.28
Average	**47.82**	45.73	44.46

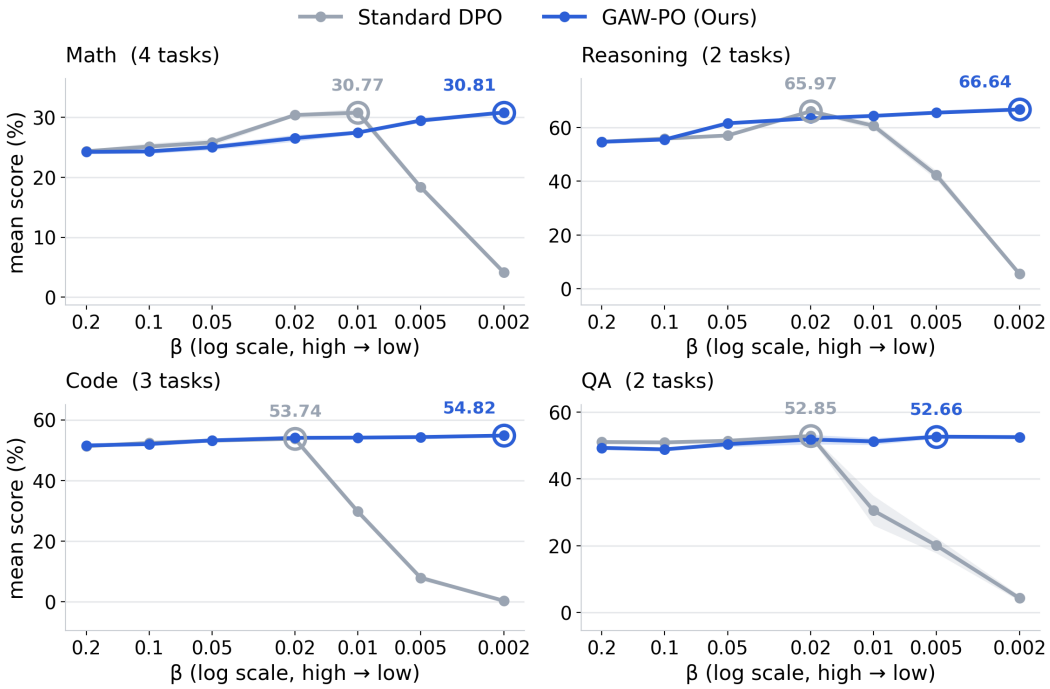

Figure 3: Performance across β grouped by benchmark category. Standard DPO degrades sharply under aggressive preference optimization across all categories, while our method remains stable and generally improves as β decreases.

Standard DPO is competitive, and occasionally better, around $\beta = 0.02$, but its performance collapses across all four categories as β becomes smaller. The effect is particularly severe for coding, reasoning, and QA, where the scores approach the bottom of their respective ranges at $\beta = 0.002$. Our method, in contrast, remains stable across all categories. Mathematics, reasoning, and coding improve consistently toward smaller values of β, while QA remains approximately stable despite the increasingly aggressive updates.

Table 5: Full benchmark results for the single dataset-level mean and clustered preferred-direction variants, all trained at $\beta = 0.002$. Values are percentages, mean ± standard deviation over 3 seeds. Bold marks the highest value in each row.

Benchmark	**Mean**	K**=2**	K**=4**	K**=8**
AIME 2024	16.77 ± 0.79	**17.67** ± 0.51	17.29 ± 1.08	16.01 ± 0.61
AIME 2025	17.53 ± 0.68	17.36 ± 0.57	**17.81** ± 0.28	17.08 ± 0.28
MATH	76.63 ± 0.39	76.81 ± 0.40	**76.98** ± 0.17	76.48 ± 0.79
Omega 500	12.33 ± 0.70	**14.20** ± 0.53	12.27 ± 0.64	12.93 ± 0.64
AGI Eval	65.49 ± 0.24	**66.37** ± 0.71	65.52 ± 0.28	65.71 ± 0.48
BBH (CoT)	**67.79** ± 0.49	67.24 ± 0.93	67.26 ± 0.59	67.57 ± 0.77
HumanEval+	75.35 ± 0.57	63.01 ± 1.40	**75.63** ± 0.79	75.61 ± 0.88
LiveCodeBench	30.01 ± 0.79	28.21 ± 0.69	30.06 ± 0.21	**30.17** ± 0.56
MBPP+	**59.11** ± 0.29	57.05 ± 0.27	58.84 ± 0.29	59.07 ± 0.74
GPQA	35.04 ± 0.39	34.90 ± 1.29	35.94 ± 2.32	**36.38** ± 1.12
MMLU (CoT)	**69.98** ± 0.57	69.79 ± 0.27	69.74 ± 0.38	69.82 ± 0.17
Average	47.82	46.60	**47.94**	47.89

E.2 Full Cluster-Variant Results

Table 5 reports the complete benchmark results corresponding to the aggregate scores shown in Table 2 of the main paper.

The results reinforce the conclusion that additional preferred directions provide only modest gains over the single dataset-level mean. $K = 4$ obtains the strongest aggregate score, while $K = 8$ is close behind and $K = 2$ performs worse. No value of K dominates across all benchmarks.

To understand this, we further measure which component attains the maximum alignment for each rejected token. In the original formulation, the paired chosen direction dominates for 79.16% of tokens, while the dataset mean dominates for 20.84%. With $K = 2$, $K = 4$, and $K = 8$, the paired direction dominates for 66.69%, 62.52%, and 53.18% of tokens, respectively, meaning that cluster-level directions determine the score for up to 46.82% of rejected tokens.

This shows that the clustered directions are actively used and capture preferred update directions not represented by the paired response alone. However, greater cluster utilization does not directly imply better downstream performance: $K = 8$ uses clustered directions most frequently but performs slightly below $K = 4$, while $K = 2$ underperforms the original formulation. Simply increasing the number of reference directions therefore does not monotonically improve the quality of the token weights; the quality of the preferred-direction decomposition matters more than the number of available directions.

F Limitations

Our method adds an offline stage that computes rejected-token weights before training. This stage is performed once per dataset and probe and is reused across all training runs; in our experiments it costs roughly 1.5 training runs. The cost is incurred once and is amortized across all subsequent training runs, hyperparameter sweeps, and ablations, as in our experiments. However, it is less lightweight than standard DPO for a single run.

Gradient alignment is measured in the parameter space of a low-rank PiSSA probe, which we use as a tractable proxy for the full parameter space. The choice is motivated by the relation between PiSSA gradients and full-parameter gradients, and the downstream results indicate that the resulting weights are informative. A systematic study of how the probe rank affects the weights, and of how closely low-rank alignment tracks full-parameter alignment, is left for future work.

The weights are computed from the SFT initialization and kept fixed during training. This matches the offline nature of DPO, where the preference pairs do not change, but the policy's gradient geom-

etry does evolve over training. Estimating the weights dynamically is an extension and is required for the online setting discussed in Section 6.

The conversion of alignment scores into weights has more possible choices. Alternative mappings are a promising direction, including temperature or offset parameters in the sigmoid, normalized reference directions, and rank- or quantile-based mappings.

Finally, our evaluation focuses on a single open model family and preference dataset, chosen to allow a controlled study, and on verifiable benchmarks in mathematics, reasoning, code, and question answering. We did not evaluate on judge-based open-ended benchmarks or on multi-turn and multilingual preference data, and extending the analysis to these settings and to larger models is an important direction for future work.