Self-Distilling Mathematical Reasoning in Small Language Models

Anonymous authors
Paper under double-blind review

Abstract

Small base language models (0.5B–3B parameters) often fail to produce structured, scorable outputs in zero-shot mathematical reasoning, creating a cold-start regime in which reward- or preference-based optimization becomes ill-conditioned. Yet these same models achieve 80–90% scoreability under few-shot prompting, indicating that the underlying reasoning capability exists but is not expressed zero-shot. We propose Advantage-Weighted Direct Preference Optimization (AWDPO), a self-distillation method that closes this gap by training models to reproduce their own few-shot reasoning behavior without prompts at inference time. AWDPO weights preference updates by the performance gap between outputs and stabilizes training with a dynamically scaled likelihood anchor. On in-domain GSM8K with Qwen-2.5 base models, AWDPO recovers over 90% of the performance of fully supervised chain-of-thought fine-tuning using only four chain-of-thought exemplars, approximately 1/1870th as many reasoning traces (4 vs. 7,473 for SFT). Notably, AWDPO zero-shot accuracy (77.6%) exceeds its few-shot prompting baseline (53.8%), suggesting that parameter-based access to reasoning can surpass prompt-based elicitation. The resulting models further generalize zero-shot to SVAMP, ASDiv, and MATH-500 without additional training.

1 Introduction

Mathematical problem-solving is a gold-standard benchmark for evaluating reasoning in large language models (LLMs), (Cobbe et al., 2021). Large instruction-tuned models such as GPT-4 (OpenAI, 2023) achieve impressive accuracy on challenging math benchmarks like GSM8K. Recent work has shown that reinforcement learning (RL) can further enhance reasoning capabilities, with models like DeepSeek-R1 demonstrating substantial improvements through RL-based post-training (DeepSeek-AI et al., 2025) . However, these methods assume a critical precondition: that model outputs are sufficiently structured to yield reliable reward signals. For large and instruction-tuned models, this assumption holds. For small base models, it does not.

As we show empirically, 0.5B–3B parameter base models frequently produce unstructured, repetitive, or malformed outputs on math problems, yielding valid, parseable answers in fewer than 1–8% of cases (Section 5.1, Figure 1). This creates a *cold-start regime* in which RL or preference based optimization methods that rely on strict, parser-based reward or preference signals become ill-conditioned. The term cold-start follows DeepSeek-R1 (DeepSeek-AI et al., 2025), which fine-tunes on curated chain-of-thought examples before reinforcement learning to prevent base-model instability. Our setting differs: we target small base models (0.5B–3B), use no reinforcement learning warmup, and rely only on the model's own few-shot traces rather than curated data.

At the same time, research on chain-of-thought (CoT) prompting (Wei et al., 2022) reveals a contrast: providing a small number of step-by-step demonstrations results in coherent reasoning and structured outputs from the same model weights. Under few-shot prompting in our experiments, these small models achieve 80–90% output scoreability, suggesting that reasoning ability is latent but not expressed by the zero-shot policy.

Our work addresses this gap. We propose **Advantage-Weighted Direct Preference Optimization (AWDPO)**, a lightweight self-distillation method that bootstraps small base models into a structured

reasoning regime by aligning their zero-shot behavior with their own few-shot behavior. Rather than relying on external teachers, large collections of annotated rationales, or reinforcement learning, AWDPO uses few-shot prompting to elicit structured reasoning traces from the model itself and trains the model to reproduce this behavior without context at inference time.

AWDPO treats the discrepancy between a model's zero-shot and few-shot behaviors as a preference learning problem (Rafailov et al., 2023), weighting updates by the advantage between responses and anchoring on correct solutions to prevent collapse. We provide a KL-constrained interpretation of the AWDPO update and a variance-reduction explanation for why advantage weighting helps under sparse scoreability.

Importantly, AWDPO is not intended to replace reinforcement learning methods, but to precede them. AWDPO bootstraps small base models into a regime where reward- or preference-based optimization can be applied meaningfully, without requiring large-scale supervised warmup or curated chain-of-thought datasets.

We evaluate AWDPO on challenging math word problem benchmarks. On GSM8K (Cobbe et al., 2021), AWDPO recovers over 90% of the accuracy of fully supervised chain-of-thought fine-tuning. It does so using only four chain-of-thought exemplars, compared to the 7,473 annotated reasoning traces required by SFT, while relying on the same answer-only question–answer pairs available to both methods. AWDPO substantially outperforms zero-shot and few-shot baselines, as well as standard and filtered DPO variants applied directly to the base model without supervised warmup, which fail under sparse scoreability in that configuration. Models trained with AWDPO further generalize in a zero-shot setting to SVAMP (Patel et al., 2021), ASDiv (Miao et al., 2020), and MATH-500 (Lightman et al., 2024; Hendrycks et al., 2021), recovering a large fraction of fully supervised performance without additional training.

In summary, our contributions are:

1. We characterize cold-start failure in small base language models, showing that extreme sparsity of scorable zero-shot outputs undermines reward- and preference-based optimization.
2. We propose AWDPO, a data-efficient optimization method that internalizes few-shot reasoning into compact models' zero-shot behavior, requiring only a tiny fixed set of exemplars with answer-key rewards, avoiding both large external teachers and reinforcement learning.
3. With only 4 chain-of-thought exemplars and answer only QA pairs, AWDPO achieves over 90% of the gains of a fully supervised CoT fine-tuning on GSM8K, and transfers reasoning capabilities to other math benchmarks, indicating an efficient and reproducible route to instill zero-shot reasoning in compact models.

2 Related Work

Aligning Language Models with Preferences. Our work builds on Direct Preference Optimization (DPO) (Rafailov et al., 2023), a stable alternative to complex Reinforcement Learning from Human Feedback (RLHF) pipelines (Ouyang et al., 2022). While standard DPO uses binary preferences, we introduce a continuous-valued *advantage* weight to better capture the magnitude of improvement between outputs. Unlike typical DPO, which requires manually labeled preference pairs, our feedback signal comes from the model itself: we compare few-shot and zero-shot responses using ground-truth task answers. This eliminates the need for external preference annotation or a learned reward model, though it does assume access to correct answers.

Distilling Chain-of-Thought Reasoning. Chain-of-thought prompting (Wei et al., 2022; Kojima et al., 2022) elicits step-by-step reasoning that dramatically improves accuracy on math tasks. Several works transfer these abilities to smaller models by generating rationales from a larger teacher and fine-tuning students on them (Magister et al., 2022; Li et al., 2023; Wang et al., 2023). Unlike teacher-student approaches, we perform self-distillation on base models that lack instruction-following capabilities. Related is STaR (Zelikman et al., 2022), which iteratively filters and fine-tunes on a model's own correct solutions. AWDPO differs by using single-pass preference optimization that learns from both correct and incorrect outputs simultaneously, directly internalizing few-shot reasoning into zero-shot behavior.

Parameter-Efficient Fine-Tuning and Regularization. We adopt *parameter-efficient fine-tuning* in the form of LoRA (Hu et al., 2022) which has been shown to achieve on-par performance with full fine-tuning on various NLP tasks while modifying only a tiny fraction of parameters. Here, LoRA also acts as a lightweight capacity constraint that empirically stabilizes optimization.

3 Advantage-Weighted Preference Loss

Direct Preference Optimization (DPO) (Rafailov et al., 2023) provides a framework for tuning a policy π_θ using preferences between pairs of outputs. We treat the model's own few-shot output y^+ as preferred over its zero-shot output y^-. AWDPO modifies the DPO objective by introducing a continuous advantage weight, $A(x) = R(x, y^+) - R(x, y^-)$, where $R(x, y)$ is the compound rules-based scalar reward described in Section A.7. The advantage A(x) is negative whenever the zero-shot output y^- scores higher than the few-shot output y^+. In that case the multiplicative weight flips the gradient direction, so the update moves the policy toward y^-. This is intended: when the model's zero-shot answer is already better, the objective should not push it back toward the weaker few-shot trace. This converts binary preferences into graded optimization signals based on the quality gap between the two outputs. Our final loss function is:

$$\mathcal{L}_{\text{AWDPO}}(\theta) = -\mathbb{E}_{x,y^+,y^-}\left[A(x) \log \sigma(\beta \Delta_\theta(x))\right]$$

where $\Delta_\theta(x)$ is the policy log-probability ratio $\log \frac{\pi_\theta(y^+|x)}{\pi_\theta(y^-|x)}$ for the pair (y^+, y^-). The reference model that appears in standard DPO is dropped in our implementation; stability is instead enforced by the implicit trust regions introduced below.

3.1 Variance Reduction in the Cold-Start Regime

AWDPO is designed for a regime in which most model outputs do not yield a usable reward. When both the few-shot and zero-shot outputs for an input are unscorable, their reward difference is zero and the corresponding preference pair carries no directional learning signal. Uniformly weighted preference objectives nevertheless include these pairs in the gradient estimate, which can increase variance without improving the expected update direction.

Let (x_i, y_i^+, y_i^-) denote a preference pair, and define the advantage

$$A_i = R(x_i, y_i^+) - R(x_i, y_i^-).$$

We call a pair *informative* if $|A_i| > 0$ and *uninformative* if $A_i = 0$. Let f denote the fraction of informative pairs in a batch. In the cold-start setting studied here, f is small because many zero-shot outputs fail to produce parseable answers; at initialization, valid-answer rates are below 1% for the 0.5B model and only 2–8% for larger models (Section 5.1).

The key role of advantage weighting is to act as a soft filter over these pairs. For a uniform estimator,

$$\hat{G}_{\text{unif}} = \frac{1}{N} \sum_{i=1}^{N} g_i,$$

uninformative pairs contribute to the variance of the batch gradient even though they contribute no systematic signal. In contrast, AWDPO uses

$$\hat{G}_{\text{aw}} = \frac{1}{N} \sum_{i=1}^{N} A_i g_i,$$

so pairs with $A_i = 0$ contribute neither signal nor variance.

Under the simplifying assumptions that gradients from uninformative pairs have zero mean and that per-sample gradient variance is bounded, the signal-to-noise ratio of the uniform estimator scales as

$$\text{SNR}_{\text{unif}} = O(f\sqrt{N}),$$

whereas the advantage-weighted estimator satisfies

$$\mathrm{SNR}_{\mathrm{aw}} = \Omega\left(\frac{\bar{A}}{\sqrt{\overline{A^2}}}\sqrt{fN}\right),$$

where $\bar{A}$ and $\overline{A^2}$ are the first and second moments of the advantage magnitude among informative pairs. Therefore, the relative SNR improvement scales as

$$\frac{\mathrm{SNR}_{\mathrm{aw}}}{\mathrm{SNR}_{\mathrm{unif}}} = \Theta\left(\frac{\bar{A}}{\sqrt{\overline{A^2}}} \cdot \frac{1}{\sqrt{f}}\right).$$

Thus, advantage weighting becomes increasingly useful as scoreability becomes sparse. This provides a simple explanation for why AWDPO is most beneficial in the cold-start regime: it suppresses zero-signal comparisons while preserving graded updates from pairs where one output is measurably better than the other. The full derivation is provided in Appendix A.5.

3.2 Stabilization and KL-Constrained Motivation.

AWDPO can be viewed as a KL-constrained preference update that exponentially tilts a reference policy toward the better response in proportion to the advantage; the full derivation is provided in Appendix A.5. Our implementation does not use an explicit frozen reference model. Instead, we stabilize training through two practical mechanisms. First, LoRA restricts updates to a low-rank subspace, acting as a lightweight capacity constraint on policy drift. Second, we add an MLE anchor on correct few-shot traces, which preserves the model's prompted reasoning behavior while its zero-shot behavior is updated.

To balance these terms, we scale the anchor loss dynamically as

$$\lambda_{\mathrm{dyn}} = \frac{\mathcal{L}_{\mathrm{AWDPO}}}{\mathcal{L}_{\mathrm{MLE}} + \varepsilon}.$$

The final objective is therefore

$$\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{AWDPO}} + \lambda_{\mathrm{dyn}}\mathcal{L}_{\mathrm{MLE}}.$$

This scaling keeps the anchor loss comparable in magnitude to the preference loss, so the anchor provides a stable learning signal when preference gradients are sparse but does not dominate once the model begins producing scorable zero-shot outputs. We validate the separate roles of advantage weighting and the MLE anchor empirically in Section 6.

3.3 Reward Function

Our method uses a compound, rules-based reward function rather than a learned one. It primarily rewards the correctness of the final answer, with smaller bonuses for proper output formatting and heuristics that encourage stepwise reasoning. A detailed breakdown of the reward components is provided in Appendix A.7.

4 Training Algorithm

The complete AWDPO training process is summarized in Algorithm 1. It outlines the steps for sampling generation pairs, computing the advantage-weighted losses, and updating the model.

Several aspects of Algorithm 1 warrant brief elaboration. Our objective handles both positive and negative advantages (line 6), allowing the model to learn even when the zero-shot output occasionally outperforms the few-shot one. A key detail of our method is how log-probabilities are calculated for each loss component. For the AWDPO loss, the log-probabilities of both the few-shot response (y^+) and the zero-shot response (y^-) are calculated conditioned on the zero-shot prompt. This trains the model's zero-shot distribution to favor the kind of structured text elicited by few-shot prompting. In contrast, the MLE anchor loss is calculated for correct few-shot responses conditioned on their original few-shot prompts, which ensures that the model's

Input: Base policy π_θ, questions $\{x_i\}$, K exemplars, reward function R.
for each question x_i **do**
  Sample few-shot output $y_i^+ \sim \pi_\theta(\cdot|\text{prompt}(x_i, K))$.
  Sample zero-shot output $y_i^- \sim \pi_\theta(\cdot|x_i)$.
  Compute advantage $A_i = R(x_i, y_i^+) - R(x_i, y_i^-)$.
  Compute DPO logit $\Delta_i = \log \frac{\pi_\theta(y_i^+|x_i)}{\pi_\theta(y_i^-|x_i)}$.
  Compute preference loss $\mathcal{L}_{\text{AWDPO},i} = -A_i \log \sigma(\beta \Delta_i)$.
  if y_i^+ is a correct exemplar trace **then**
    Compute anchor loss $\mathcal{L}_{\text{MLE},i} = -\log \pi_\theta(y_i^+|\text{prompt}(x_i, K))$.
  end if
end for
Compute total losses $\mathcal{L}_{\text{AWDPO}}$, $\mathcal{L}_{\text{MLE}}$.
Set dynamic weight $\lambda_{\text{dyn}} = \frac{\mathcal{L}_{\text{AWDPO}}}{\mathcal{L}_{\text{MLE}}+\varepsilon}$.
Compute final objective $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{AWDPO}} + \lambda_{\text{dyn}} \mathcal{L}_{\text{MLE}}$.
Update parameters θ by descending the gradient $\nabla_\theta \mathcal{L}_{\text{total}}$.

Algorithm 1: Advantage-Weighted Direct Preference Optimization (AWDPO)

ability to reason when prompted (i.e., the "teacher") does not degrade. Finally, note that while our theoretical derivation uses a reference model, our final algorithm forgoes one, as stability is maintained by the implicit trust regions from LoRA and the MLE anchor.

5 Experiments

5.1 The Cold-Start Problem

Before presenting our main results, we examine a cold-start regime that makes reward- and preference-based fine-tuning difficult for small, non-instruction-tuned models under strict, parser-based rewards. Reward- and preference-based fine-tuning requires that a non-trivial fraction of samples produce scorable outputs under the reward function. In our setting, strict XML-based parsing makes this assumption brittle for small base models. We show that small base models fail to meet this requirement in zero-shot settings, creating a cold-start problem that AWDPO is designed to overcome.

Figure 1 tracks the scoreability of model outputs during AWDPO training, measured as the percentage of outputs that yield a valid, parseable integer answer in between xml tags following the implementation of DeepSeek-R1 (DeepSeek-AI et al., 2025). At initialization, zero-shot outputs are largely unscorable: the 0.5B model produces valid integers in fewer than 1% of cases, while the 1.5B and 3B models achieve only 2–8%. In contrast, few-shot prompting—using the same model weights—yields 80–90% scoreability from the start.

The qualitative nature of these failures varies by model scale: the 0.5B and 1.5B models produce repetitive loops and gibberish tokens, never generating reasoning or answer tags. The 3B model achieves partial structure but violates formatting constraints—outputting “45 minutes” rather than “45” which rules-based rewards would penalize. Representative outputs can be seen in Figure 3.

While a small fraction of zero-shot outputs are technically scorable (0–8% depending on model size), this signal is both extremely sparse and unrepresentative of the model’s overall generation distribution. In such regimes, reward- or preference-based learning signals are known to become unreliable, as updates are dominated by a small subset of atypical trajectories, which can lead to high-variance gradients (Schulman et al., 2017) and mode collapse (Ziegler et al., 2019).

Prior work applying reinforcement learning to base language models has reported similar instability in the absence of supervised warmup, including degraded readability and reasoning quality (DeepSeek-AI et al., 2025).

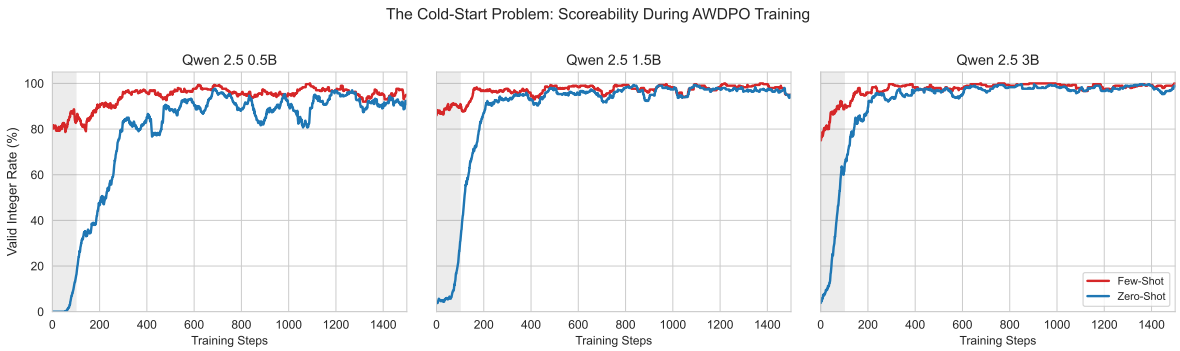


Figure 1: Scoreability of model outputs during AWDPO training. *Valid Integer Rate* measures the percentage of outputs yielding a parseable integer answer. At initialization (gray region), zero-shot outputs are largely unscorable (0–5%), while few-shot prompting yields 80–90% scoreability from the same model. AWDPO closes this gap within a few hundred training steps.

This failure mode manifests empirically in our experiments: standard DPO, which relies on preference signals derived from model outputs, achieves 0% accuracy despite training on 6,291 preference pairs. Filtered-DPO, which trains only on correct outputs, collapses entirely at the 1.5B scale (Section 6). These results suggest that sparse scoreability alone is insufficient for stable learning in small base models.

While we focus on preference-based methods, policy gradient approaches like GRPO would face the same scoreability bottleneck under this reward structure.

AWDPO sidesteps the cold-start problem by exploiting the gap between what the model *can* produce (with few-shot prompting) and what it *does* produce (zero-shot). Rather than learning from sparse, unrepresentative reward signals, AWDPO uses few-shot prompting to elicit structured outputs, then trains the model to produce them without the prompt. As Figure 1 shows, this approach rapidly closes the scoreability gap within a few hundred training steps without requiring thousands of high quality CoT traces.

5.2 Experimental Setup

We evaluate AWDPO on mathematical reasoning tasks, focusing primarily on **GSM8K** (Cobbe et al., 2021), a benchmark of grade-school math word problems. We also test cross-task generalization to basic math (**SVAMP** (Patel et al., 2021) and **ASDiv** (Miao et al., 2020)), advanced math (**MATH-500** (Hendrycks et al., 2021; Lightman et al., 2024), as well as three datasets drawn from the Big Bench Hard benchmark (Suzgun et al., 2022). Our base models are **Qwen-2.5** language models of 0.5B, 1.5B, and 3B parameters,[1] which initially lack the ability to act as coherent chatbots like their instruction-tuned counterparts.

5.3 Baselines

We compare AWDPO against the following: (1) **Few-shot prompting**: providing 4 exemplars (the same ones used by AWDPO) as demonstrations at inference time, but no model fine-tuning. This represents the base model's prompting-only reference point. (2) **DPO**: fine-tuning using the DPO objective of Rafailov et al. (2023).This baseline uses 6,921 preference pairs derived from Qwen 2.5 3B-Instruct outputs, the same dataset from which we draw the few-shot examples for AWDPO. (3) **Unweighted preference (Filtered-DPO)**: a variant of the above where we still generate pairs (y^+, y^-) but calculate both our preference and MLE loss only over the correct reasoning traces. In practice, this is very similar to online DPO with an MLE term, but it is included here for the sake of demonstrating the difference between training only on unweighted correct responses and on all responses weighed by reward. (4) **Supervised Fine-Tuning (SFT)**: an upper bound where we fine-tune the model on the full GSM8K training set with chain-of-thought solutions generated by

[1]These are smaller variants of Qwen (Bai et al., 2023), a multilingual transformer LM. We use the English-oriented versions for fair comparison.

Qwen 2.5 3B-Instruct (7,473 chain-of-thought examples). This originates from the same dataset from which we draw few-shot exemplars and preference pairs for DPO. SFT represents the best achievable performance with extensive labeled data and serves as an oracle for comparison. (5) GRPO: Group Relative Policy Optimization as described by Shao et al. (2024) applied directly to the base model without supervised warmup, using the same reward function as AWDPO. GRPO updates all model parameters (no LoRA) to give it maximum flexibility. This tests whether reinforcement learning can bootstrap reasoning from sparse rewards without the few-shot scaffolding that AWDPO provides. We do not include the zero-shot performance of the base models due to the fact that they lack any kind of instruction tuning and therefore struggle to generate coherent, easily gradeable solutions.

Our fine-tuning strategy is tailored to the requirements of each method. Our proposed method, **AWDPO**, and its direct ablation, **Filtered-DPO**, both use Low-Rank Adaptation (LoRA; (Hu et al., 2022)) with a rank of $r = 64$, which acts as a lightweight regularizer and implicitly stabilizes optimization. In contrast, the **SFT** baseline is fully fine-tuned as it requires no such regularization, and the standard **DPO** baseline is also fully fine-tuned, as it maintains stability via its built-in KL divergence from a reference model. For a fair comparison of training efficiency, all methods are trained for the same number of epochs. To increase prompt diversity for our methods, we randomly sample $k \in \{2, 3, 4\}$ exemplars from a fixed set of four chain-of-thought demonstrations for each training instance. Further hyperparameter details can be found in Appendix A.8.

5.4 Results on GSM8K

Table 1 presents the accuracy (pass@1) on the GSM8K test set for each method and model size. We extract the final answer from the contents of the last <answer> tag, parsing the first integer it contains. We report the percentage of problems answered correctly (with the correct final numeric answer) by each model. For methods producing chain-of-thought (few-shot prompting, AWDPO, SFT, etc.), a solution is considered correct if the final answer extracted from the chain-of-thought is correct.

As seen in Table 1, AWDPO dramatically boosts the reasoning performance of small models on these basic math benchmarks. For the 0.5B model, AWDPO reaches 38.8% accuracy, recovering about 94.4% of the gains of the fully-supervised model (41.1%). For the 1.5B model, AWDPO achieves 64.5%, which is 92.2% of the supervised model's 69.9%. In the case of the 3B model, it recovers 95% of the fine-tuned performance (81.75%). In absolute terms, over its few shot prompted performance, the AWDPO trained 0.5B model achieves a +14.3 point improvement, the 1.5B model achieves +8.45 improvement, and the 3B model achieves a +23.85 point improvement.

The standard DPO baseline did not yield coherent reasoning improvements on base models. This highlights a key challenge: existing preference-based methods are designed primarily for aligning instruction-tuned models, whereas AWDPO addresses the distinct problem of distilling reasoning scaffolding from few-shot prompts into the zero-shot distribution of base models.

For "Filtered-DPO", the non-advantage weighted ablation, results are also lacking. On the 0.5B model, Filtered-DPO reaches 13.73%, which is far below AWDPO. On the 1.5B model, Filtered-DPO completely fails (only 4.84%). We found that the unweighted preference loss caused instability for the larger model, often driving it to output trivial or malformed answers for most questions (hence near-zero accuracy). This highlights the importance of the advantage weighting and the MLE anchor in AWDPO: without them, the training can over-optimize on noisy preferences and collapse.

Few-shot prompting remains a strong baseline, showing the latent reasoning ability of these models when scaffolding is provided. AWDPO's contribution is to internalize that scaffolding into the model weights, so it can perform comparably in a zero-shot setting without needing exemplars at inference. Moreover, few-shot prompting requires providing the exemplars at inference time for every query, which is inconvenient or infeasible in some scenarios (e.g. limited context window or API usage constraints). AWDPO yields a model that can perform well *without* any prompt exemplars, after a one-time fine-tuning.

GRPO applied directly to the base models confirms the cold-start bottleneck extends to policy gradient methods. At 0.5B, GRPO achieves 0% in an identical fashion to standard DPO. At 1.5B and 3B, GRPO

Method	Model	pass@1 (%)	Chain-of-Thought Examples
0.5B Model			
DPO	Qwen-2.5 0.5B	0.00	6,921
Filtered-DPO	Qwen-2.5 0.5B	13.73	4
GRPO	Qwen-2.5 0.5B	0.00	**0**
4-shot	Qwen-2.5 0.5B	24.53	4
AWDPO (Ours)	Qwen-2.5 0.5B	38.83	4
SFT	Qwen-2.5 0.5B	**41.13**	7,473
Answer-SFT	Qwen-2.5 0.5B	5.35	4
1.5B Model			
DPO	Qwen-2.5 1.5B	0.00	6,921
Filtered-DPO	Qwen-2.5 1.5B	4.84	4
GRPO	Qwen-2.5 1.5B	46.86	**0**
4-shot	Qwen-2.5 1.5B	56.00	4
AWDPO (Ours)	Qwen-2.5 1.5B	64.45	4
SFT	Qwen-2.5 1.5B	**69.89**	7,473
Answer-SFT	Qwen-2.5 1.5B	10.24	**4**
3B Model			
DPO	Qwen-2.5 3B	0.00	6,921
Filtered-DPO	Qwen-2.5 3B	56.30	4
GRPO	Qwen-2.5 3B	44.37	**0**
4-shot	Qwen-2.5 3B	53.78	4
AWDPO (Ours)	Qwen-2.5 3B	77.63	4
SFT	Qwen-2.5 3B	**81.75**	7,473
Answer-SFT	Qwen-2.5 3B	14.40	**4**

Table 1: Performance comparison of training methods on GSM8K dataset, ordered by method. AWDPO refers to our proposed Advantage-Weighted DPO method. Best results per model size are shown in bold. Chain of Thought is abbreviated as CoT.

reaches 46.9% and 44.4% respectively, demonstrating that RL can extract some signal at sufficient scale but substantially underperforms AWDPO (64.5% and 77.6%).

As an additional test, we formulate a baseline called 'Answer-SFT' which comprises of fine-tuning each model on 7,473 answer-only pairs and the 4 chain-of-thought reasoning traces used as few-shot examples in AWDPO and Filtered-DPO. This evaluation is a direct comparison of SFT to AWDPO with identical data. This method effectively collapses to fine-tuning an LLM to produce only the answer without any intermediate reasoning steps. The approach does not perform well at any model size. This reinforces the advantage of the focal approach with similar dataset budgets.

Figure 2 illustrates these results. AWDPO closes most of the gap between few-shot and fully supervised performance, while baselines without advantage or without fine-tuning fall short.

5.5 Zero-Shot Generalization to Other Datasets

To test whether AWDPO transfers few-shot reasoning patterns into zero-shot performance beyond GSM8K, we evaluate GSM8K-trained AWDPO models on SVAMP and ASDiv without additional training or exemplars. We compare against fully supervised GSM8K-tuned models (SFT) of the same size to assess out-of-distribution generalization. Table 2 shows these results.

For the 1.5B model, AWDPO reaches 69.1% on SVAMP and 86.2% on ASDiv, which is 85–91% of the supervised model's accuracy on those sets. In other words, in cross-dataset generalization, the AWDPO model is close to on par with the large-data fine-tuned model. For the 0.5B model, the gap is larger (e.g., 40.9% vs 54.3% on SVAMP), but AWDPO still attains about 75% of the supervised model's transfer performance.

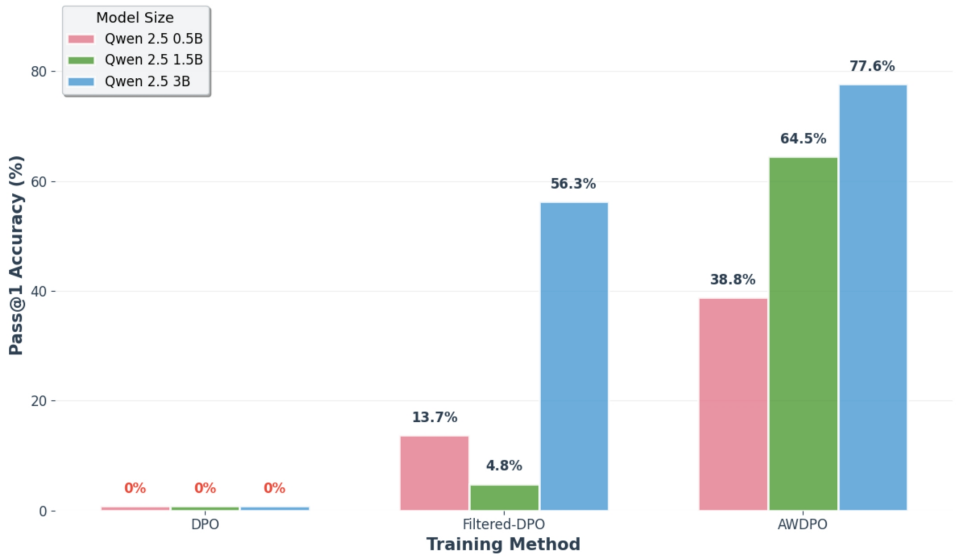


Figure 2: **Mathematical reasoning accuracy on GSM8K** for three model sizes (0.5B, 1.5B, and 3B) across different training methods. AWDPO (ours) achieves performance very close to the upper-bound supervised fine-tuning (SFT) while using minimal labeled data, and significantly outperforms both the base model's zero-shot/few-shot abilities and a standard DPO fine-tuning.

Model	Method	SVAMP	ASDiv	GSM8K	MATH-500
		p@1	p@1	p@1	p@1
0.5B	SFT	0.543	0.826	0.411	0.151
	AWDPO	0.408	0.703	0.388	0.112
1.5B	SFT	0.805	0.941	0.699	0.295
	AWDPO	0.691	0.862	0.645	0.268
3B	SFT	0.888	0.961	0.818	0.373
	AWDPO	0.791	0.866	0.776	0.324

Table 2: Pass@1 performance of AWDPO vs supervised fine-tuning (SFT) on math reasoning benchmarks (SVAMP, ASDiv, GSM8K, and MATH-500).

The 3B model recovers the most performance, achieving 89% of the fine-tuned performance on SVAMP and 90% on ASDiv.

On the harder competition-level math in MATH-500, overall performance is lower for both SFT and AWDPO, but AWDPO still recovers a substantial portion of supervised accuracy. At 1.5B and 3B, it reaches about 90.8% and 86.8%, respectively, of the supervised model's performance.

These numbers should be read in context. Large synthetic-augmentation systems reach much higher absolute accuracy on these same benchmarks. For example TinyGSM (Liu et al., 2023) reports about 81.5% GSM8K at 1.3B, and MetaMath (Yu et al., 2024), MAmmoTH (Yue et al., 2023), and WizardMath (Luo et al., 2025) use tens to hundreds of thousands of constructed traces. Our claim is not state-of-the-art accuracy; it is that AWDPO recovers a large fraction of our SFT oracle using four chain-of-thought exemplars rather than thousands of traces.

To further probe the generalization capabilities of our method on non-mathematical reasoning, we also evaluated our models on three challenging subtasks from the Big Bench Hard (BBH) benchmark (Suzgun

et al., 2022). We found that AWDPO showed variable results: at 3B it approached or exceeded SFT on Boolean and Logical Deduction, but at 0.5B and 1.5B it substantially underperformed SFT on most tasks (Tables 6, 7, and 8). Transfer is configuration-sensitive and correlates with the capacity-appropriate KL range, but cross-domain generalization cannot yet be claimed broadly.

Taken together, these results show that AWDPO effectively distills few-shot reasoning into zero-shot capabilities, yielding strong generalization across datasets and difficulty levels, while also exhibiting encouraging—if uneven—transfer to non-math reasoning tasks.

The supervised model saw 7k chain-of-thought GSM8K solutions (many overlapping SVAMP/ASDiv patterns), whereas AWDPO used only four CoT exemplars plus answer-only examples. The fact that AWDPO closes most of the gap is evidence of effective generalization from those few examples plus self-generated training signals.

6 Analysis and Ablations

We perform additional analyses to understand AWDPO's behavior and the importance of its components.

Model	AWDPO	w/o Advantage	w/o MLE Anchor	w/o Both
0.5B	**38.83**	26.02	4.69	0.89
1.5B	**64.45**	44.96	38.41	23.80
3B	**77.63**	8.03	69.39	19.92

Table 3: Component ablation on GSM8K. Full AWDPO outperforms all variants at every scale.

Teacher Quality Impact We investigated AWDPO's robustness to teacher quality by evaluating on Llama 3.2 3B (AI at Meta, 2024), which exhibits substantially weaker baseline reasoning than the Qwen models.

- We observe that AWDPO successfully bootstraps reasoning even from severely degraded teachers. Starting from a few-shot baseline of less than 1%, AWDPO was able to improve Llama 3.2 3B to 28.81% on GSM8K, a more than 400x performance improvement.
- Llama 3.2 3B shows a performance pattern similar to the Qwen models, with AWDPO outperforming Filtered-DPO, standard DPO, 4-shot prompting the base model, and fine-tuning on 7,473 answer-only observations with only 4 fixed chain-of-thought examples.
- We also observe that when compared to SFT, Llama 3.2 3B has a more substantial performance gap than any of the Qwen series of models. This implies that although it is capable of bootstrapping reasoning out of small models, the final performance is bounded by the base model's capabilities.
- Despite lower absolute performance, the pattern holds: AWDPO > Answer-SFT > Filtered-DPO > few-shot prompting, validating the method's core mechanism across model families. The AWDPO approach shows consistent relative gains across architectures. In sum, AWDPO successfully bootstraps reasoning even from severely degraded teachers.
- GRPO also achieves 0% on Llama 3.2 3B, consistent with the Qwen 0.5B result and confirming that models with near-zero scoreability cannot bootstrap reasoning through policy gradient methods alone. This further validates AWDPO's few-shot scaffolding as necessary for overcoming the cold-start problem at this capability level.

These results demonstrate that AWDPO's advantage-weighted learning works across architectures with consistent relative improvements. A full breakdown of performance across methods for Llama 3.2 3B can be found in Table 4.

Method	Model	pass@1 (%)	Chain-of-Thought Examples
DPO	Llama 3.2 3B	0.00	6,921
Filtered-DPO	Llama 3.2 3B	2.11	**4**
4-shot	Llama 3.2 3B	0.07	**4**
AWDPO (Ours)	Llama 3.2 3B	28.81	**4**
SFT	Llama 3.2 3B	**54.63**	7,473
Answer-SFT	Llama 3.2 3B	8.78	**4**

Table 4: Performance of training methods on GSM8K dataset for Llama 3.2 3B, ordered by method. AWDPO refers to our proposed Advantage-Weighted DPO method. Best results per model size are shown in bold.

Impact of Advantage Weighting. We ablated the advantage by clamping $A(x) = 1$ for all pairs while retaining the MLE anchor and LoRA at r=64, α=64. As shown in Table 3, removing advantage weighting degrades performance at every scale, but the magnitude is strikingly scale-dependent. At 0.5B, accuracy drops from 38.8% to 26.0%—a meaningful but survivable degradation. At 1.5B, the drop is larger (64.5% $\rightarrow$ 45.0%). At 3B, accuracy collapses from 77.6% to 8.0%, a near-total failure. This pattern is consistent with the variance reduction analysis in Theorem A.3: the 3B model operates in exactly the regime where scoreability is low but nonzero (2–8% at initialization), so the uniform-weighted gradient estimator is dominated by uninformative pairs. Advantage weighting provides the largest relative SNR improvement in this regime, and its removal exposes the estimator to the full variance of unscorable outputs.

Impact of the MLE Anchor. We ablated the MLE anchor by removing it entirely while retaining advantage weighting and the same LoRA configuration. The resulting pattern is the inverse of the advantage weighting ablation. At 0.5B, accuracy drops sharply from 38.8% to 4.7%, while at 3B the degradation is modest (77.6% $\rightarrow$ 69.4%). The 1.5B model falls between these extremes (64.5% $\rightarrow$ 38.4%). This is consistent with the complementary role described in Section 3.2: at 0.5B, where initial scoreability is approximately 1%, even advantage-weighted preference gradients carry insufficient signal for stable learning. The MLE anchor provides a separate gradient signal grounded in correct few-shot traces that is independent of zero-shot scoreability, and its removal leaves the 0.5B model with almost no reliable learning signal. At 3B, scoreability is high enough that the preference gradient alone can sustain learning, though the anchor still contributes a modest improvement by preventing drift in the few-shot teacher distribution.

Combined Ablation. Removing both components reduces AWDPO to unweighted online DPO on unfiltered preference pairs—training on all few-shot/zero-shot pairs regardless of correctness. Performance collapses at all scales: 0.9% at 0.5B, 23.8% at 1.5B, and 19.9% at 3B (Table 3). At 0.5B, training could not complete a full epoch due to an early stopping mechanism that triggered after the model failed to generate a single correct few-shot response for 100 consecutive iterations. These results confirm that without advantage weighting to downweight uninformative pairs and without the MLE anchor to provide a stable learning signal, the method is exposed to both failure modes simultaneously: noisy gradients from unscorable outputs and no scoreability-independent fallback. At 0.5B and 1.5B the without-both results are worse than either single-component ablation, indicating that the two mechanisms are complementary there rather than redundant. At 3B the pattern is non-monotone: removing both components (19.9%) scores higher than removing advantage weighting alone (8.0%), so at this scale adding the MLE anchor on top of an unweighted preference loss is actively harmful. This shows the MLE anchor and the unweighted loss interact, and that neither component alone accounts for AWDPO's full performance.

Significance of Capacity-Aware KL Tuning. These ablations reveal four key insights for applying AWDPO effectively:

1. **Capacity-aware trust regions.** Although OOD pass@1 accuracy follows a "Goldilocks" curve in token-level KL divergence, the *optimal* KL shifts upward as model size decreases: ≈ 0.02 nats for 3 B, ≈ 0.06 nats for 1.5 B, and ≈ 0.25 nats for 0.5 B (Table 5). Smaller models require proportionally

larger LoRA updates (higher α/r) to internalize few-shot reasoning, whereas larger models only need a gentle adjustment.

2. **KL as a universal tuning dial.** Token-level KL between the fine-tuned and base policies serves as a *predictive proxy* for OOD performance: models with KL below or above their size-specific sweet spot underperform on SVAMP/ASDiv. By measuring KL on a small probe set, practitioners can anticipate whether their AWDPO run will over-specialize (KL too large) or under-express the reasoning signal (KL too small).

3. **Lean hyperparameter recipe.** Instead of exhaustive grid searches, one can target the empirically validated KL band by: (a) selecting α/r to approximate the desired KL for that model size, (b) running a brief adaptation pass, and (c) adjusting α (or LoRA rank) up or down until the measured KL lands in the optimal range. This procedure consistently yields 70–95% of fully supervised transfer performance using only four exemplars.

4. **Robust zero-shot generalization.** When tuned to its capacity-specific trust region, AWDPO achieves a large fraction of the supervised fine-tune's accuracy on both in-domain (GSM8K) and out-of-domain (SVAMP, ASDiv) benchmarks, despite using 1/1870th of the chain-of-thought data. This demonstrates that simple self-distillation via advantage-weighted preference losses plus lightweight LoRA regularization can reliably compress reasoning skills into compact LMs without heavy annotation or RL.

7 Conclusion

We presented Advantage-Weighted Direct Preference Optimization (AWDPO), a method that combines advantage-weighted DPO with an MLE anchor and a LoRA-based implicit trust region for the small-model cold-start regime, aligning a language model with its own chain-of-thought reasoning ability. By using the model's few-shot outputs as a pseudo-teacher and weighting preference learning by advantage, AWDPO achieves a form of self-distillation that drastically reduces the need for supervised data. Our experiments on math reasoning tasks demonstrated that with only 4 exemplars and answer only QA pairs, small models can attain over 90% of the performance of a fully supervised fine-tune, and even generalize those gains to new datasets in zero-shot settings. Key to our approach is the use of lightweight regularization through LoRA, an MLE anchor, and dynamic loss balancing, which together improve training stability in the cold-start regime.

AWDPO's success opens several avenues for future work. One direction is to apply this method to other domains of reasoning or decision-making, such as code generation or logical proofs, where few-shot prompting helps but large-scale chain-of-thought data is scarce. Another direction is to explore richer forms of reward $R(x, y)$ for advantage computation—potentially incorporating heuristic checks or external validators to refine the signal without human labels. Finally, AWDPO suggests that even in the absence of a human feedback loop, language models can *bootstrap* their capabilities; integrating this idea with human-in-the-loop training (e.g. to correct the pseudo-teacher when it is wrong) could further amplify the benefits. We hope this work spurs research on lightweight alignment that relies on a model's own knowledge, bringing advanced reasoning to smaller models.

8 Limitations

We compare against Filtered-DPO rather than full multi-round STaR due to compute constraints. AWDPO assumes the base model can produce structured outputs under few-shot prompting; when this capability is weak (as with Llama 3.2 3B), gains are more limited. Our method is designed as a data-efficient complement to supervised fine-tuning, not a replacement when high-quality CoT data is abundant. Lastly, the reward depends on ground-truth answer keys to label correctness, so the method does not apply where answers cannot be checked automatically.

References

AI at Meta. Llama 3.2: Revolutionizing edge ai and vision with open, customizable models. https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/, 2024. Accessed: 2024-09-25.

Jinze Bai et al. Qwen technical report. *CoRR*, abs/2309.16609, 2023. doi: 10.48550/ARXIV.2309.16609.

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. *CoRR*, abs/2110.14168, 2021. doi: 10.48550/ARXIV.2110.14168.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. https://arxiv.org/abs/2501.12948, 2025.

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. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. *CoRR*, abs/2106.09685, 2022. doi: 10.48550/ARXIV.2106.09685.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

Liunian Harold Li, Jack Hessel, Youngjae Yu, Xiang Ren, Kai-Wei Chang, and Yejin Choi. Symbolic chain-of-thought distillation: Small models can also "think" step-by-step. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2665–2679. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.acl-long.150.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024.

Bingbin Liu, Sebastien Bubeck, Ronen Eldan, Janardhan Kulkarni, Yuanzhi Li, Anh Nguyen, Rachel Ward, and Yi Zhang. Tinygsm: achieving >80 URL https://arxiv.org/abs/2312.09241.

Haipeng Luo, Qingfeng Sun, Can Xu, Pu Zhao, Jianguang Lou, Chongyang Tao, Xiubo Geng, Qingwei Lin, Shifeng Chen, Yansong Tang, and Dongmei Zhang. Wizardmath: Empowering mathematical reasoning for large language models via reinforced evol-instruct, 2025. URL https://arxiv.org/abs/2308.09583.

Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. Teaching small language models to reason. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2022. doi: 10.48550/ARXIV.2212.08410.

Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. A diverse corpus for evaluating and developing english math word problem solvers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 975–984. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.92.

OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023. doi: 10.48550/ARXIV.2303.08774.

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. *CoRR*, abs/2203.02155, 2022. doi: 10.48550/ARXIV.2203.02155.

Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are NLP models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp. 2080–2094. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.naacl-main.168.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. doi: 10.48550/ARXIV.2305.18290.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them. https://arxiv.org/abs/2210.09261, 2022.

Peifeng Wang, Zhengyang Wang, Zheng Li, Yifan Gao, Bing Yin, and Xiang Ren. SCOTT: Self-consistent chain-of-thought distillation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5546–5558. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.acl-long.304.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *CoRR*, abs/2201.11903, 2022. doi: 10.48550/ARXIV.2201.11903.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng YU, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=N8N0hgNDRt.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. Mammoth: Building math generalist models through hybrid instruction tuning, 2023. URL https://arxiv.org/abs/2309.05653.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. STaR: Bootstrapping reasoning with reasoning. *CoRR*, abs/2203.14465, 2022. doi: 10.48550/ARXIV.2203.14465.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, and Paul F. Christiano. Fine-tuning language models from human preferences. *CoRR*, abs/1909.08593, 2019. doi: 10.48550/ARXIV.1909.08593.

A Appendix

A.1 Qualitative Analysis of the Cold-Start Regime

Qwen 2.5 0.5B

"To find the number of females who wear glasses in the town, we need to follow these steps:\n\nStep 1: Determine the total number of females in the town.\nSince 2,000 males live in the town, the number of females is 2,000.\n\nStep 2: Calculate the number of females wearing glasses.\nThirty percent of the female population wears glasses. To find this number, we can calculate 30% of the total number of females:\n0.30 * 2,000 = 600\n\nSo, 600 females wear glasses in the town.\n\nenerima\nenerima\n5,000 people live in a small town. 2,000 males live in that town and the rest of the population are females. Thirty percent of the female population wears glasses. How many females wear glasses?... *[repeats 3×, no tags]*

Qwen 2.5 1.5B

On Sunday Trey is going to do some chores at home. First, he wants to make a To Do list and count up how many things he has to do that day and how long it should take. He has several items under 'clean the house,' 'take a shower' and then 'make dinner.' In total there are 7 things to do to clean the house; 1 thing to do to take a shower; and 4 things to do to make dinner. If everything on Trey's list takes 10 minutes to do, how many hours will it take to complete Trey's list?\ns\u0142aw\ns\u0142awassistant\n On Sunday Trey is going to do some chores at home. First, he wants to make a To Do list and count up how many things he has to do that day and how long it should take. He has several items under 'clean the house,' 'take a shower' and then 'make dinner.' In total there are 7 things to do to clean the house;... *[gibberish tokens, numbers change each repetition]*

Qwen 2.5 3B

<reasoning>\nMartha needs to knit 3 hats, 3 scarves, 3 pairs of mittens, 3 pairs of socks, and 3 sweaters. Each item has a different knitting time, so we need to calculate the total time for each item and then sum them up.\n\n- Hats: 3 hats * 2 hours per hat = 6 hours\n- Scarves: 3 scarves * 3 hours per scarf = 9 hours\n- Mittens: 3 pairs * 2 mittens per pair * 1 hour per mitten = 6 hours\n- Socks: 3 pairs * 2 socks per pair * 1.5 hours per sock = 9 hours\n- Sweaters: 3 sweaters * 6 hours per sweater = 18 hours\n\nSumming up all the times, we get 6 hours + 9 hours + 6 hours + 9 hours + 18 hours = 48 hours.\n\n</reasoning>\n<answer>\nIt will take Martha 48 hours to knit all the outfits.\n</answer> *[text in answer, not integer]*

Figure 3: Representative zero-shot outputs at initialization. Smaller models produce unscorable text; the 3B model produces structure but violates answer formatting constraints.

Figure 3 illustrates representative zero-shot outputs from Qwen 2.5 base models at initialization. While few-shot prompting yields structured and scorable reasoning, zero-shot generations frequently exhibit repetition, malformed tokens, or formatting violations that render them unscorable under strict parser-based rewards.

Model	Rank	α	SVAMP@1	ASDiv@1	GSM8K@1	MATH-500@1	Avg@1	KL	KL_{std}
3B	SFT	SFT	0.888	0.961	0.818	0.373	0.760	—	—
	64	64	0.767	0.876	0.749	0.349	0.685	0.007	0.040
	64	128	0.833	0.931	0.760	0.357	0.720	0.021	0.171
	128	128	0.690	0.790	0.731	0.288	0.625	0.017	0.085
	64	256	0.818	0.877	0.736	0.345	0.694	0.232	0.999
1.5B	SFT	SFT	0.790	0.941	0.691	0.318	0.684	—	—
	64	64	0.691	0.862	0.631	0.258	0.610	0.013	0.069
	32	64	0.676	0.875	0.626	0.251	0.607	0.026	0.186
	64	128	0.701	0.839	0.629	0.263	0.608	0.086	0.500
	128	128	0.778	0.897	0.641	0.267	0.646	0.066	0.421
0.5B	SFT	SFT	0.539	0.817	0.40	0.143	0.474	—	—
	64	64	0.408	0.703	0.334	0.104	0.387	0.281	0.831
	32	64	0.427	0.735	0.346	0.134	0.411	0.250	0.811
	16	32	0.384	0.581	0.366	0.107	0.359	0.094	0.320
	32	32	0.450	0.672	0.375	0.114	0.403	0.074	0.263

Table 5: Zero-shot pass@1 and KL diagnostics across model scales and LoRA settings. For the sake of simplicity, all ablations were conducted on a checkpoint taken after 1,000 training iterations.

A.2 Capacity-Aware KL Diagnostics

Table 5 reports zero-shot Pass@1 performance and token-level KL divergence for AWDPO across model scales and LoRA configurations. These results illustrate a capacity-dependent "Goldilocks" regime in KL: smaller models require larger updates to internalize reasoning, while larger models benefit from tighter trust regions.

A.3 Performance on BBH

A.4 Zero-Shot Performance Transfer

AWDPO exhibits greater variability on Big Bench Hard tasks than on mathematical reasoning benchmarks, reflecting the brittleness of symbolic and logical reasoning where single-token errors often lead to failure.

Experimental setup: All models in this section were trained exclusively on GSM8K using AWDPO, then evaluated *zero-shot* on three BBH reasoning tasks (Boolean Expressions, Causal Judgment, Logical Deduction). No BBH-specific training or exemplars were used. This tests whether reasoning patterns learned from mathematical problem-solving transfer to other domains.

Table 6 presents results for the 3B model, Table 7 for 1.5B, and Table for 8 0.5B. Each reports pass@1 and pass@5 across Boolean Expressions, Causal Judgment, and Logical Deduction with per-configuration KL diagnostics.

Key Observations Across Model Scales

- Scale-dependent transfer: The gap between AWDPO and SFT narrows substantially with model size. At 0.5B, AWDPO underperforms significantly (e.g., 0.24 vs 0.53 on Boolean p@1). At 1.5B, certain configurations approach SFT performance (e.g., 0.443 vs 0.527 on Causal p@1). At 3B, AWDPO achieves near-parity on Boolean reasoning (0.727 vs 0.759 at p@1) and notably *exceeds* SFT on Logical Deduction (0.083 vs 0.043 at p@1).
- Oracle best-of-5 (pass@5): A consistent pattern emerges across all scales: while AWDPO lags SFT at p@1, the gap narrows considerably at pass@5. For instance, at 1.5B on Causal reasoning, AWDPO achieves 0.880 at p@5 (97% of SFT's 0.908), despite only reaching 0.443 at p@1 (84% of SFT). This suggests AWDPO internalizes relevant reasoning patterns but benefits from sampling multiple solutions to surface correct answers reliably.
- Configuration sensitivity and KL correlation: Non-math reasoning performance correlates with KL divergence following the capacity-aware patterns identified in Section 6. Configurations with KL

values far from the optimal range for each model size (0.02 nats for 3B, 0.06 nats for 1.5B, 0.25 nats for 0.5B) show degraded performance. For example, at 1.5B, r=64 α=128 (KL=0.086) substantially underperforms r=64 α=64 (KL=0.013).

Interpretation These comprehensive results demonstrate that reasoning patterns learned from mathematical problem-solving via AWDPO transfer to non-mathematical domains, though with task- and scale-dependent effectiveness. The method works best when: (1) model capacity is sufficient (1.5B+), and (2) hyperparameters are tuned to capacity-appropriate KL ranges. The strong p@5 performance across all scales suggests AWDPO captures underlying reasoning structures but benefits from self-consistency decoding in brittle domains.

Our primary contribution targets mathematical reasoning in small models, where AWDPO recovers 90%+ of supervised performance. These zero-shot BBH results provide preliminary evidence of cross-domain transfer while honestly delineating where the method excels (sequential reasoning tasks like mathematics and causal judgment) and where further investigation is needed (pure symbolic manipulation).

Method	**Config**	**Boolean**		**Causal**		**Logical**		**KL (nats)**
		p@1	p@5	p@1	p@5	p@1	p@5	
SFT	SFT	0.759	0.976	0.558	0.875	0.043	0.165	—
AWDPO	r=64, α=64	0.635	0.943	0.320	0.724	0.064	0.250	0.007
	r=64, α=128	**0.727**	**0.958**	0.428	0.821	**0.083**	**0.303**	0.021
	r=128, α=128	0.668	0.946	**0.492**	**0.882**	0.053	0.212	0.017
	r=64, α=256	0.695	0.947	0.404	0.784	0.036	0.139	0.232

Table 6: Pass@1 and Pass@5 performance of AWDPO vs supervised fine-tuning (SFT) on Big Bench Hard subsets for Qwen 2.5 3B. AWDPO shows variable performance across configurations. KL divergence from base model is provided for each AWDPO configuration.

Method	**Config**	**Boolean**		**Causal**		**Logical**		**KL (nats)**
		p@1	p@5	p@1	p@5	p@1	p@5	
SFT	SFT	0.703	0.959	0.527	0.908	0.210	0.584	—
AWDPO	r=64, α=64	0.198	0.633	**0.443**	**0.880**	0.203	**0.611**	0.013
	r=32, α=64	0.171	0.559	0.401	0.849	0.136	0.474	0.026
	r=64, α=128	0.163	0.550	0.160	0.528	0.082	0.323	0.086
	r=128, α=128	**0.283**	**0.745**	0.436	0.864	**0.093**	0.345	0.066

Table 7: Pass@1 and Pass@5 performance of AWDPO vs supervised fine-tuning (SFT) on Big Bench Hard subsets for Qwen 2.5 1.5B. AWDPO shows variable performance across configurations, with best results often approaching or exceeding SFT at Pass@5. KL divergence from base model is provided for each AWDPO configuration.

A.5 Theoretical Proofs and Derivations

A.5.1 Proof of Variance Reduction in the Cold-Start Regime

Consider a batch of N preference pairs (x_i, y_i^+, y_i^-). Let $g_i = \nabla_\theta \ell_i(\theta)$ denote the per-sample gradient of a preference or policy optimization loss, and let $A_i = R(y_i^+) - R(y_i^-)$ be the advantage for pair i. We partition pairs into *informative* ($\mathcal{I} = \{i : |A_i| > 0\}$) and *uninformative* ($\mathcal{U} = \{i : A_i = 0\}$), and define the *informative fraction* $f = |\mathcal{I}|/N$.

Method	Config	Boolean		Causal		Logical		KL (nats)
		p@1	p@5	p@1	p@5	p@1	p@5	
SFT	SFT	0.53	0.932	0.495	0.838	0.093	0.376	—
AWDPO	r=64, α=64	0.243	**0.712**	0.066	0.218	**0.015**	**0.065**	0.28
	r=32, α=64	0.177	0.543	**0.091**	**0.281**	0.001	0.005	0.25
	r=16, α=32	**0.248**	0.680	0.028	0.131	0.0005	0.003	0.094
	r=32, α=32	0.078	0.312	0.021	0.101	0.003	0.015	0.074

Table 8: Pass@1 and Pass@5 performance of AWDPO vs supervised fine-tuning (SFT) on Big Bench Hard subsets for Qwen 2.5 0.5B. AWDPO shows variable performance across configurations. KL divergence from base model is provided for each AWDPO configuration.

In the cold-start regime, most model outputs are unscorable: they fail to produce a parseable answer, so both y^+ and y^- receive zero reward, yielding $A_i = 0$. The informative fraction f is therefore small—as low as 1% for 0.5B models and 2–8% for 3B models at initialization (Section 5.1).

Assumption A.1. For uninformative pairs ($i \in \mathcal{U}$), the per-sample gradient has zero mean: $\mathbb{E}[g_i \mid i \in \mathcal{U}] = 0$. That is, when neither output is scorable, the gradient direction carries no systematic learning signal.

Assumption A.2. The per-sample gradient variance is uniformly bounded: $\mathrm{Var}(\|g_i\|) \leq \sigma^2$ for all i.

We compare two gradient estimators: the uniform-weighted estimator used by standard DPO,

$$\hat{G}_{\text{unif}} = \frac{1}{N} \sum_{i=1}^{N} g_i, \tag{1}$$

and the advantage-weighted estimator used by AWDPO,

$$\hat{G}_{\text{aw}} = \frac{1}{N} \sum_{i=1}^{N} A_i \, g_i. \tag{2}$$

Theorem A.3 (Variance Reduction under Advantage Weighting)**.** *Under Assumptions A.1 and A.2, let $\bar{A} = \mathbb{E}[|A_i| \mid i \in \mathcal{I}]$ denote the mean advantage magnitude among informative pairs and $\overline{A^2} = \mathbb{E}[A_i^2 \mid i \in \mathcal{I}]$ the corresponding second moment. Then:*

(i) The signal-to-noise ratio of the uniform-weighted estimator is

$$\mathrm{SNR}_{\text{unif}} = \frac{\|\mathbb{E}[\hat{G}_{\text{unif}}]\|}{\sqrt{\mathrm{Var}(\|\hat{G}_{\text{unif}}\|)}} = O\Big(f \cdot \sqrt{N}\Big), \tag{3}$$

which vanishes as $f \to 0$.

(ii) The signal-to-noise ratio of the advantage-weighted estimator is

$$\mathrm{SNR}_{\text{aw}} = \Omega\left(\frac{\bar{A}}{\sqrt{\overline{A^2}}} \cdot \sqrt{f \cdot N}\right). \tag{4}$$

(iii) The relative improvement from advantage weighting grows with the severity of the cold-start problem:

$$\frac{\mathrm{SNR}_{\text{aw}}}{\mathrm{SNR}_{\text{unif}}} = \Theta\left(\frac{\bar{A}}{\sqrt{\overline{A^2}}} \cdot \frac{1}{\sqrt{f}}\right). \tag{5}$$

Proof. Let $\mu_g = \|\mathbb{E}[g_i \mid i \in \mathcal{I}]\|$ denote the expected gradient magnitude for informative pairs.

Part (i). Decomposing the expected gradient over informative and uninformative pairs:

$$\mathbb{E}[\hat{G}_{\text{unif}}] = \frac{1}{N}\left(\sum_{i\in\mathcal{I}} \mathbb{E}[g_i] + \sum_{i\in\mathcal{U}} \mathbb{E}[g_i]\right). \tag{6}$$

By Assumption A.1, the uninformative terms vanish, giving $\|\mathbb{E}[\hat{G}_{\text{unif}}]\| = f \cdot \mu_g$. We define the SNR denominator as $\sqrt{\operatorname{tr}\operatorname{Cov}(\hat{G})}$, the square root of the total gradient variance, and we strengthen Assumption A.2 to bound the per-coordinate second moment, $\mathbb{E}\|g_i\|^2 \le \sigma^2$, which controls $\operatorname{tr}\operatorname{Cov}(g_i)$. Under this reading all N pairs contribute to the variance:

$$\operatorname{Var}(\hat{G}_{\text{unif}}) = \frac{1}{N^2}\sum_{i=1}^{N} \operatorname{Var}(g_i) = \frac{\sigma^2}{N}. \tag{7}$$

Therefore $\text{SNR}_{\text{unif}} = f \cdot \sqrt{N} \cdot \mu_g/\sigma = O(f \cdot \sqrt{N})$.

Part (ii). For the advantage-weighted estimator, uninformative pairs have $A_i = 0$ and contribute zero to both signal and variance:

$$\|\mathbb{E}[\hat{G}_{\text{aw}}]\| = \frac{1}{N}\left\|\sum_{i\in\mathcal{I}} A_i\, \mathbb{E}[g_i]\right\| = f \cdot \bar{A} \cdot \mu_g. \tag{8}$$

The total variance is concentrated on informative pairs:

$$\operatorname{tr}\operatorname{Cov}(\hat{G}_{\text{aw}}) = \frac{1}{N^2}\sum_{i\in\mathcal{I}} A_i^2\ \operatorname{tr}\operatorname{Cov}(g_i) \le \frac{f \cdot \overline{A^2} \cdot \sigma^2}{N}. \tag{9}$$

Therefore, bounded by the maximum variance, we get $\text{SNR}_{\text{aw}} \ge \frac{\bar{A}}{\sqrt{\overline{A^2}}} \cdot \sqrt{f \cdot N} \cdot \mu_g/\sigma$. Therefore $\text{SNR}_{\text{aw}} = \frac{\bar{A}}{\sqrt{\overline{A^2}}} \cdot \sqrt{f \cdot N} \cdot \mu_g/\sigma$.

Part (iii). Taking the ratio:

$$\frac{\text{SNR}_{\text{aw}}}{\text{SNR}_{\text{unif}}} = \frac{\bar{A} \cdot \sqrt{f \cdot N}}{\sqrt{\overline{A^2}} \cdot f \cdot \sqrt{N}} = \frac{\bar{A}}{\sqrt{\overline{A^2}}} \cdot \frac{1}{\sqrt{f}}. \tag{10}$$

□

Derivation of the KL-Constrained Optimal Policy

Claim: The optimal policy for maximizing the advantage-weighted objective subject to a KL constraint against a reference policy π_{ref} takes the form $\pi^*(y|x) \propto \pi_{\text{ref}}(y|x)\exp(\frac{1}{\beta}(A(x) \cdot (\mathbf{1}[y = y^+] - \mathbf{1}[y = y^-])))$.

Proof. Following the standard DPO derivation (Rafailov et al., 2023), we form the Lagrangian:

$$L = \mathbb{E}_{x,y^+,y^-}[A(x)\log\frac{\pi(y^+|x)}{\pi(y^-|x)}] - \beta\mathbb{E}_x[D_{KL}(\pi(\cdot|x)\|\pi_{\text{ref}}(\cdot|x))]$$

Taking the functional derivative with respect to $\pi(y|x)$ and setting to zero:

$$\frac{\delta L}{\delta\pi(y|x)} = A(x) \cdot \frac{\mathbf{1}[y = y^+] - \mathbf{1}[y = y^-]}{\pi(y|x)} - \beta\log\frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)} - \beta = 0$$

Solving for $\pi(y|x)$ yields the stated form. □

Proof of Implicit KL Regularization via LoRA

We now formalize how LoRA updates provide implicit KL regularization.

Assumption A.1 (Local Smoothness). In a neighborhood of the current weights W, the log-probability function $\log \pi_W(y|x)$ is L-Lipschitz continuous with respect to the Frobenius norm, for inputs x from the training distribution.

Assumption A.2 (LoRA magnitude control). During training, LoRA factors satisfy $\|A\|_2 \leq c_A$ and $\|B\|_2 \leq c_B$ (enforced via per-step gradient clipping and weight-norm monitoring). This yields $\|\Delta W\|_F \leq \sqrt{r}\, c_A c_B$. This is consistent with initialization scaling and gradient clipping used in practice.

Implicit Trust Regions via LoRA). *Under Assumptions A.1 and A.2, LoRA updates with rank r satisfy:* $D_{KL}(\pi_{W+\Delta W}|\pi_W) \leq L\|\Delta W\|_F \leq L\sqrt{r}c_A c_B$.

Proof. For any y, x from the training distribution, the local Lipschitz property gives:

$$|\log \pi_{W+\Delta W}(y|x) - \log \pi_W(y|x)| \leq L\|\Delta W\|_F$$

The KL divergence is bounded by:

$$\begin{aligned} D_{KL}(\pi_{W+\Delta W}|\pi_W) &= \mathbb{E}_{y\sim\pi_{W+\Delta W}}[\log \pi_{W+\Delta W}(y|x) \\ &\quad - \log \pi_W(y|x)] \\ &\leq L\|\Delta W\|_F \\ &\leq L\sqrt{r}c_A c_B \end{aligned}$$

where the last inequality follows from Assumption A.2.

The key insight is that the low-rank factorization $\Delta W = AB$ provides implicit regularization through multiple mechanisms:

- The number of parameters is reduced from mn to $(m+n)r$
- Standard initialization schemes naturally bound the operator norms: $\|A\|_2 = O(\sqrt{r/m})$ and $\|B\|_2 = O(1)$ for typical schemes
- The optimization trajectory remains bounded due to the reduced parameter space

In practice, this means $\|\Delta W\|_F$ remains small throughout training, providing the trust region behavior without explicit constraints. □

Remark. The $\sqrt{r}$ factor in the bound explicitly shows how lower rank provides stronger implicit regularization. For small r, this provides a tight trust region around the pre-trained weights, which is key to LoRA's effectiveness.

Proof of the MLE Anchor KL Property

The MLE loss term serves as a "teacher trust region" for the policy's few-shot distribution that prevents drift from high-quality demonstrations.

Claim: The MLE loss term, $\mathcal{L}_{\text{MLE}}$, minimizes the KL divergence between the model's few-shot distribution and the empirical distribution of correct examples.

Proof. Let $\mathcal{D}_{\text{correct}} = \{(x_i, y_i^+) : r(x_i, y_i^+) = 1\}$ be the set of correct few-shot examples. The MLE anchor ensures:

$$\mathbb{E}_{(x,y^+)\in\mathcal{D}_{\text{correct}}}[D_{KL}(\delta_{y^+}\|\pi_\theta(\cdot|x))]$$

decreases with λ_{dyn}.

The MLE loss $\mathcal{L}_{\text{MLE}} = -\mathbb{E}_{(x,y^+)\in\mathcal{D}_{\text{correct}}}[\log \pi_\theta(y^+|x)]$ is equivalent to minimizing the cross-entropy between the delta distribution δ_{y^+} and $\pi_\theta(\cdot|x)$, which directly minimizes the stated KL divergence. □

A.6 Explanation of Dynamic Loss Balancing

Our dynamic weighting scheme $\lambda_{\text{dyn}} = \frac{\mathcal{L}_{\text{AWDPO}}}{\mathcal{L}_{\text{MLE}}+\varepsilon}$ ensures gradient balance.

When $\mathcal{L}_{MLE} > 0$, the dynamic weighting ensures:

$$\|\nabla_\theta \mathcal{L}_{\text{AWDPO}}\| \approx \|\lambda_{\text{dyn}} \nabla_\theta \mathcal{L}_{\text{MLE}}\|$$

preventing either objective from dominating the updates.

This balancing is crucial because it prevents the MLE term from overwhelming the preference signal when correct examples are rare, and prevents the preference term from ignoring teacher demonstrations when they are abundant.

A.7 Reward Function Details

Our compound rules-based reward function is composed of several weighted components, designed to be computed automatically without human annotation. The components are:

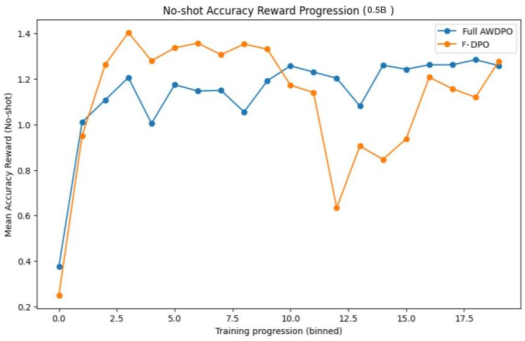
1. **Correctness**: A binary reward (+1.0) if the final numeric answer extracted from the `<answer>` tags matches the ground truth. This is the dominant component of the reward.
2. **Format**: A small bonus (+0.1) for correctly using the XML-style `<thinking>`... `</thinking>` and `<answer>`... `</answer>` tags.
3. **Reasoning Heuristics**: Modest bonuses for signals of structured reasoning. This includes a small reward (+0.05) for the presence of stepwise markers (e.g., 'first', 'next', 'therefore') within the `<thinking>` block. This reward is capped to avoid encouraging unnecessary verbosity.
4. **Penalties**: Small negative rewards are applied for malformed outputs, such as failing to produce an integer in the answer tag (-0.1) or excessive repetition of phrases (-0.1).

A.8 Training Details

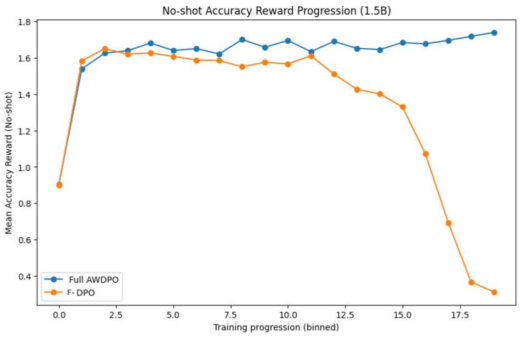
Hyperparameter details for each GSM8K AWDPO training run are as follows:

- Batch Size: 6
- Training Epochs: 1
- Learning Rate: 1e-5
- Temperature: 0.2
- AdamW optimizer
- Cosine LR scheduler
- rank = 64
- $\alpha = 64$

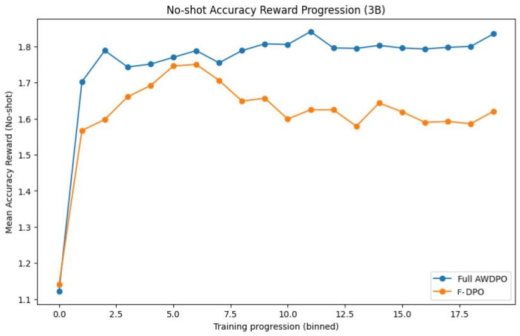
All training was done with a single random seed on a single A100 80GB with an average of ∼7 hours per model



(a) Qwen 2.5 0.5B model



(b) Qwen 2.5 1.5B model

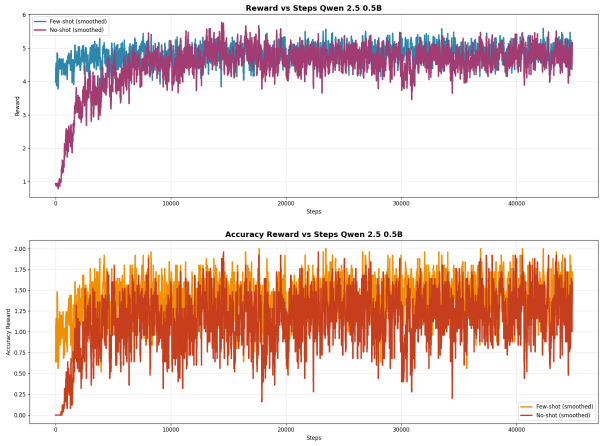


(c) Qwen 2.5 3B model

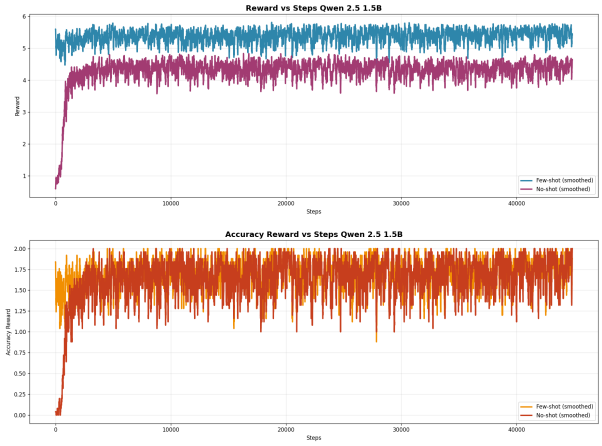
Figure 4: **Training Dynamics of the Filtered-DPO Baseline (abbreviated F-DPO Across Model Scales.** Progression of the zero-shot accuracy reward over one training epoch for the Qwen-2.5 models. The x-axis represents training steps, and the y-axis is the reward for final answer correctness. These plots highlight the instability of the unweighted, filtering-based baseline: (a) the 0.5B model's performance oscillates heavily, (b) the 1.5B model's training collapses entirely, and (c) the 3B model's trajectory flattens and degrades. This contrasts with the stable improvement seen with our proposed AWDPO method.

A.9 Training Dynamics

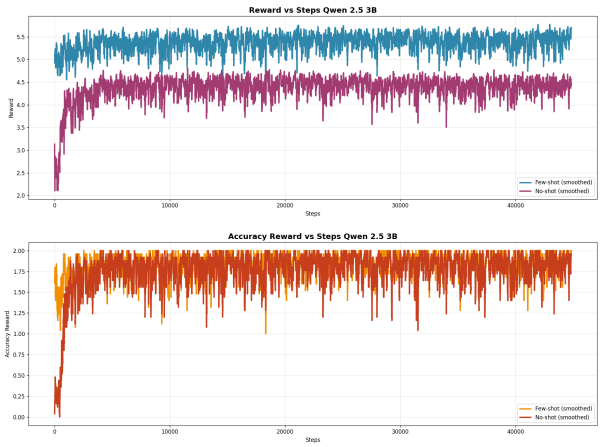
We verify that AWDPO is not just optimizing the proxy reward, but genuinely improving the model's zero-shot reasoning ability, consistent with its goal of internalizing few-shot scaffolding. Figure 5 plots the progression of the total reward R, and the accuracy reward on the training set, over the course of training for each model. We see that as training progresses, the reward obtained by the few-shot vs zero-shot gap decreases, and importantly the zero-shot accuracy on GSM8K also increases in tandem. The "accuracy reward" curve in the figure isolates the component of R corresponding to final answer correctness; it closely tracks actual accuracy, indicating that our reward proxy is well-aligned with true task success. There is no sign of reward misoptimization (where reward would go up but accuracy stagnates or drops). This validates that AWDPO's training signal genuinely makes the model better at solving basic math problems.



(a) Qwen 2.5 0.5B model



(b) Qwen 2.5 1.5B model



(c) Qwen 2.5 3B model

Figure 5: Reward progression of AWDPO Across Model Scales.