SHOULD SFT PREFER THE FAMILIAR? DIVERSITY COLLAPSE AND RECOVERY VIA MODEL MERGING

Anonymous authors
Paper under double-blind review

ABSTRACT

Supervised fine-tuning (SFT) is widely used for downstream adaptation and often serves as initialization for further reinforcement learning (RL). Recently, several methods have improved SFT by upweighting tokens that are already familiar to the current policy. Some approaches such as DFT (Wu et al., 2026) prioritize high-probability tokens within external data, while some approaches train on self-generated trajectories, which naturally contain tokens that are familiar to the current model. Although these methods have good post-SFT performance, we empirically find that preferring familiar tokens in SFT can substantially reduce policy diversity and lead to worse subsequent RL performance. We further show theoretically that such methods create a self-reinforcing dynamic that progressively contracts the model's effective support, causing the model to become overly concentrated and reduce the diversity. To mitigate this effect, we propose a simple recovery approach: merging the fine-tuned model with its pretrained base model before RL. This approach can restore diversity lost during fine-tuning while preserving the gains in post-SFT performance. Across multiple mathematical reasoning and code generation benchmarks, base-model merging consistently outperforms recent SFT variants and standard regularization approaches such as KL-divergence regularization, yielding improvements in both post-SFT accuracy and subsequent RL performance.

1 INTRODUCTION

Supervised fine-tuning (SFT) is one of the most widely used approaches for adapting pretrained language models to downstream tasks. In modern post-training pipelines, the SFT checkpoint also serves as the starting point for further reinforcement learning (RL). This gives SFT two distinct goals: it should produce a strong model on its own, while preserving enough diversity for the model to continue improving through RL.

Standard SFT minimizes token-level cross-entropy on fixed external demonstrations, assigning every target token the same explicit weight. Recent works have found that low-probability tokens can induce large noisy updates due to noisy information caused by low-probability tokens in external data, and focusing on target tokens that the model already has high probability can be beneficial for SFT performance (Wu et al., 2026) and for mitigating catastrophic forgetting (Chen et al., 2025). For example, DFT (Wu et al., 2026) and TALR (Lin et al., 2026) assign larger weights to tokens with higher probabilities under the current model, while ProFit (Liu et al., 2026b) masks tokens below a probability threshold. RAFT (Dong et al., 2023; Xiong et al., 2025) and Iterative-SFT (Chen et al., 2025) implement the same preference at the data level by fine-tuning on self-generated responses, which naturally contain more tokens that already have high probability under the current model. These methods share a common trend: training more on familiar, high-probability tokens can improve SFT performance.

However, post-SFT performance is not the only objective: an SFT checkpoint should also support strong performance after subsequent RL. We find that the gains from preferring familiar tokens come with a huge and previously overlooked cost: *diversity collapse*. Taking DFT (Wu et al., 2026) as an example, Figure 1 reveals a clear trade-off. DFT achieves higher post-SFT accuracy but sharply reduces the diversity and decrease Pass@8. Hence, as shown in Figure 1, the improvement in DFT

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.

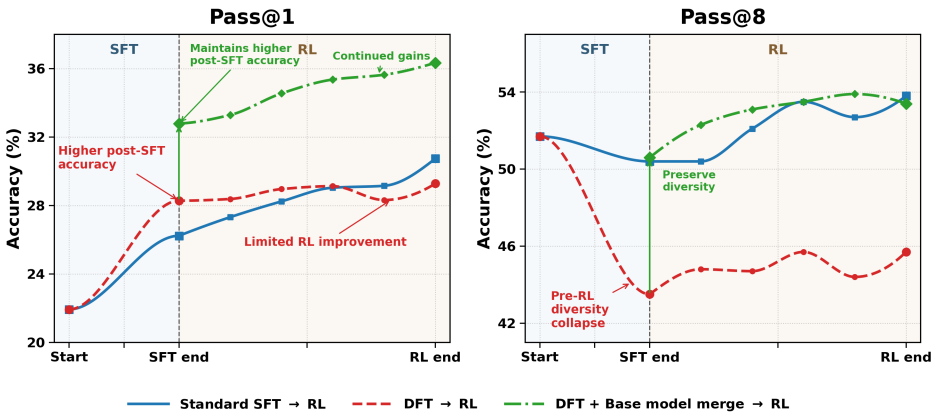


Figure 1: Diversity collapse and recovery in DFT, an SFT variant that favors high-probability tokens. We use Qwen3-4B-Base and report the average Pass@1 and Pass@8 accuracy over OlympiadBench. DFT improves accuracy but leads to diversity collapse, limiting subsequent RL gains. Base model merging approach restores diversity while maintaining and even improving the high accuracy from DFT, enabling continued improvement during RL and best final performance.

through RL is limited. In contrast, standard SFT achieves lower post-SFT accuracy but preserves the diversity and finally outperforms DFT after RL for both Pass@1 and Pass@8. We therefore ask:

Does preferring familiar tokens in SFT lead to diversity collapse and worse RL performance? If so, how can we retain its accuracy benefits while preserving the model's ability to improve through subsequent RL?

We first answer the former question empirically and theoretically. Empirically, we conduct experiments that continuously vary the degree of preference on familiar tokens, in three representative algorithms: DFT, ProFit and RAFT-style algorithm. As this preference becomes stronger, we observe a higher Pass@1 accuracy but consistently lower entropy and diversity, which also limits the gains from subsequent RL training. To explain these observations, we develop a unified theoretical framework that characterizes standard SFT and recent improvements over SFT that prefer the familiar tokens. Our analysis reveals a fundamental difference in the optimization dynamics. The standard SFT is driven toward matching the external data distribution and does not lead to a severe diversity collapse at the population level. In contrast, preferring familiar tokens that are already preferred by the current policy further increases their probability in subsequent updates, biasing the model away from the external data distribution used for fine-tuning. Within our framework, we show that many representative algorithms like DFT, ProFit and TALR reduce the effective support of the policy, leading to a substantial loss of policy diversity. This establishes that diversity collapse is a consequence of preferring familiar tokens during SFT.

Motivated by this understanding, we next investigate how to retain the accuracy benefits of preferring the familiar while mitigating diversity collapse. We identify a simple yet effective approach: merging the fine-tuned model with its pretrained base model before subsequent RL. The base model contains diversity that has been suppressed during fine-tuning, and merging it with the fine-tuned model can help to recover this lost diversity and significantly increase Pass@k. Moreover, the merged model can also achieve higher Pass@1, potentially by combining the task-specific capabilities from the fine-tuned model with the broader capabilities of the base model. Hence, the merged model consistently achieves the strongest performance over SFT and its variants, in both accuracy and subsequent RL performance on multiple math and coding benchmarks.

In summary, our main contributions are:

- We identify a previously underexplored trade-off in SFT methods that prefer familiar tokens: doing so improves immediate Pass@1 performance but induces substantial diversity collapse and degrades subsequent RL performance.

- We empirically show that preferring familiar tokens is a key driver of diversity collapse: strengthening this preference across SFT variants consistently improves Pass@1 while reducing entropy and answer diversity, finally limiting the gains from subsequent RL training.
- We provide a unified theoretical framework of standard SFT and recent variants where familiar tokens are preferred. While standard SFT is matching the external data distribution, we show that other variants progressively reduce the effective support of the policy.
- We show that merging the fine-tuned model with its pretrained base model is a simple yet effective approach that can both retain the accuracy gains of recent SFT variants and also preserve room for subsequent RL improvement. Compared with many recent SFT variants and regularization approaches like KL-divergence regularization, base-model merging preserves and in some cases improves the accuracy gains obtained by preferring familiar tokens while recovering diversity and substantially improving subsequent RL performance.

2 Related Works

Supervised Fine-Tuning and Variants. Supervised Fine-Tuning (SFT) is one of the most common approaches for post-training language models. Given a fixed set of external demonstrations, standard SFT minimizes the token-level cross-entropy loss to imitate the target responses. Recent works have improved SFT from both the optimization and data perspectives. From the optimization perspective, a line of work makes the training objective more aware of the model's own distribution, for example by assigning larger weights to more familiar tokens or suppressing poorly supported ones (Wu et al., 2026; Liu et al., 2026b; Lin et al., 2026; Wang et al., 2026; Zhang et al., 2026c), introducing additional regularization or uncertainty-aware weighting to mitigate distributional drift and forgetting (Zhu et al., 2026; Diao et al., 2026), or deriving importance-weighted objectives from an RL or off-policy perspective (Qin and Springenberg, 2025; Zhang et al., 2026b). From the data perspective, GRAPE (Zhang et al., 2026a) selects externally generated trajectories that are more likely under the target model, while self-improvement approaches like RAFT (Dong et al., 2023) and Iterative-SFT (Chen et al., 2025) repeatedly regenerate the training data from the model itself. We provide a more detailed discussion of these methods in Appendix A.1.

Diversity Collapse and Capability Preservation in Post-Training. Prior works have shown that continued or excessive SFT can improve immediate performance while reducing diversity or model plasticity. Compared to them, our work investigates recent SFT variants that train on model-familiar tokens, and shows that such objectives can substantially accelerate diversity collapse even within a single epoch. Previous works by Yuan et al. (2025); Wu et al. (2025) observe diversity collapse and degraded out-of-domain generalization during iterative self-improvement. Our work further studies whether familiarity bias is a key driver of diversity collapse and whether merging only at the end can improve the diversity. Similar base model merging at the end has also been considered in earlier works (Dang et al., 2025; Liu et al., 2026a; Yuan et al., 2025) that investigate mitigating excessive SFT. Diversity collapse has also been widely observed during RL itself, where rapidly decreasing policy entropy can limit exploration (Yu et al., 2026; Cui et al., 2025; Jiang et al., 2025; Hao et al., 2026), motivating methods that improve diversity or Pass@k during RL (Li et al., 2026; Cui et al., 2025; Tajwar et al., 2026; Peng et al., 2025).

3 Preliminaries

Supervised Fine-Tuning A language model parameterized by θ defines a conditional distribution $\pi_\theta(y \mid x) = \prod_{t=1}^{T} \pi_\theta(y_t \mid x, y_{<t})$, where x is an input prompt and $y = (y_1, \ldots, y_T)$ is the corresponding response. We denote the pretrained model by π_0 and its fine-tuned version by π_θ. Given a supervised dataset $\mathcal{D}$ of prompt–response pairs (x, y), Supervised Fine-Tuning (SFT) aims to minimize the negative log-likelihood

$$\mathcal{L}_{\mathrm{SFT}}(\theta) = -\mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\sum_{t=1}^{T} \log \pi_\theta(y_t \mid x, y_{<t}) \right].$$

Preference on Familiar Tokens in SFT. Several recent works improve SFT performance by training more on high-probability tokens under the current model. Some of them assign larger weights

to high-probability target tokens, hence learning more from tokens already familiar to the current model. We write this class of objectives as

$$\mathcal{L}_\omega(\theta) = -\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\sum_{t=1}^{T} \mathrm{sg}(\omega_\theta(x, y_{\le t}))\log \pi_\theta(y_t \mid x, y_{<t})\right], \tag{1}$$

where ω_θ is a detached weighting function computed from the current policy. Different algorithms correspond to different choices of ω_θ. For example, SFT uses $\omega_\theta = 1$, DFT (Wu et al., 2026) sets $\omega_\theta = \pi_\theta(y_t \mid x, y_{<t})$, ProFit (Liu et al., 2026b) uses $\omega_\theta = \mathbb{I}\{\mathrm{sg}(\pi_\theta(y_t \mid x, y_{<t})) \ge \tau\}$, TALR (Lin et al., 2026) uses $\omega_\theta = \max\{\omega_{\min}, \mathrm{sg}(\pi_t^{1/\delta})\}$ and changes δ based on the batch loss.

Some other works such as RAFT (Dong et al., 2023) and Iterative-SFT (Chen et al., 2025) change the training data itself to address limited coverage and forgetting. Throughout this paper, we refer to iterative methods that perform SFT on filtered, self-generated responses as *Iterative Rejection Sampling* (IRS). At iteration k, for a batch of N prompts, responses are generated by the current policy: $y \sim \pi_{\theta_k}(\cdot \mid x)$. The generated responses are then filtered according to reward to form the dataset $\mathcal{D}_k$ for iteration k. The model is then optimized on $\mathcal{D}_k$ using the standard SFT objective.

Despite their different implementations, these two lines of work share a common preference: they prefer tokens already familiar to the current model. DFT, ProFit, and TALR do so explicitly by assigning greater relative weight to high-probability tokens, while IRS-style algorithms do so implicitly: tokens with high probability under the current policy are more likely to appear in its self-generated responses and thus to appear in the training data. As we show in the following sections, this shared tendency creates a self-reinforcing dynamic that increasingly concentrates the policy, ultimately leading to diversity collapse.

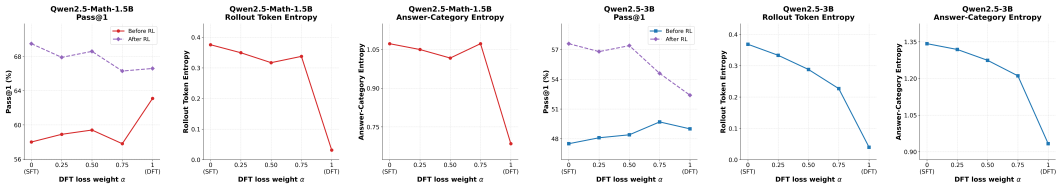
4 Preferring Familiar Tokens Leads to Diversity Collapse

We study three different SFT variants that prefer the familiar tokens: continuous token reweighting in DFT, hard token filtering in ProFit, and training with self-generated data in IRS. Although they are different approaches, they all have more training signals placed on tokens that are more familiar under the current policy. This allows us to test whether the observed diversity collapse is a common consequence of this familiarity-biased learning. For DFT algorithm with $\omega_\theta = \pi_\theta(y_t \mid x, y_{<t})$ in Eq. 1, we combine the DFT and SFT objectives using a coefficient α to vary the strength of this preference:

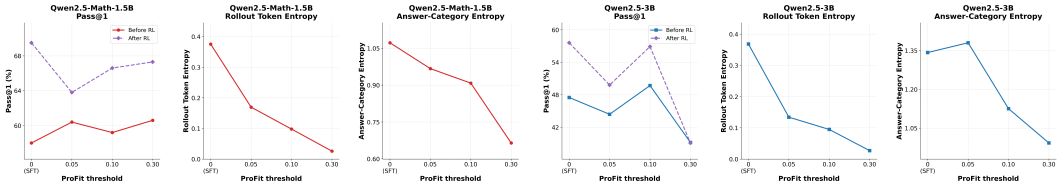
$$\omega_{\alpha,\theta}(y_t \mid x, y_{<t}) = \alpha \cdot \pi_\theta(y_t \mid x, y_{<t}) + (1-\alpha)\cdot 1 \tag{2}$$

and use $\omega_{\alpha,\theta}$ for token weights. The induced loss can also be regarded as a linear combination of the two losses $\mathcal{L}_{w_\alpha}(\theta) = \alpha\cdot\mathcal{L}_{\mathrm{DFT}}(\theta) + (1-\alpha)\cdot\mathcal{L}_{\mathrm{SFT}}(\theta)$. A larger α indicates a stronger preference for familiar tokens, with $\alpha = 0$ corresponding to pure SFT and $\alpha = 1$ to pure DFT. For ProFit algorithm with $\omega_{\tau,\theta} = \mathbb{I}\{\mathrm{sg}(\pi_\theta(y_t \mid x, y_{<t}) \ge \tau\}$ in Eq. 1, we set $\tau \in \{0, 0.05, 0.1, 0.3\}$, while $\tau = 0$ corresponds to standard SFT. A larger τ places greater emphasis on familiar tokens. For IRS with self-generated data (Dong et al., 2023; Yuan et al., 2025; Chen et al., 2025), at each stage, the current policy samples 16 responses for each of N prompts. If any response is correct, we select one correct response as the training data point. The resulting filtered dataset is then used for the next optimization stage. Because tokens favored by the current policy are more likely to appear in its own rollouts, training on these responses implicitly learns more from familiar tokens. A smaller rollout interval N refreshes the rollout policy more frequently, keeping the training data closer to the current model distribution and hence strengthening this preference. We vary N to study how the strength of this preference affects the model.

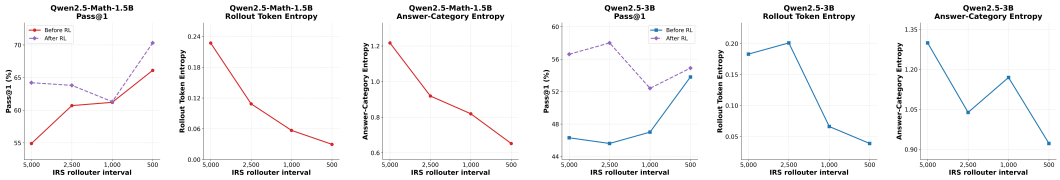
We train Qwen2.5-Math-1.5B and Qwen2.5-3B on 10K samples from NuminaMath-CoT (Li et al., 2024) for all experiments. We evaluate them on the MATH-OAI test set. For each problem, we sample 16 responses. We first measure performance before RL and after RL using Pass@1, and measure generation diversity with two metrics: *mean rollout token entropy* and *answer-category entropy*. Mean rollout token entropy is the token entropy averaged over all generated tokens and rollouts, which captures local uncertainty during generation. Answer-category entropy measures answer-level diversity by computing the entropy of the final-answer distribution across 16 rollouts for each prompt and then averaging it over all prompts. In Appendix D.6, we also show that strengthening the preference on the familiar tokens generally decreases Pass@8.



(a) We interpolate between the SFT and DFT token weights using coefficient α in Eq. 2. A larger α indicates a stronger preference for familiar and high-probability tokens of the current model.



(b) We vary the threshold τ in ProFit. A larger τ indicates a stronger preference for familiar and high-probability tokens of the current model.



(c) We vary the rollout interval in IRS. Smaller interval refreshes the rollout policy more frequently, making the training data potentially contain more tokens that are familiar to the current model.

Figure 2: **Accuracy and diversity under varying strengths of preference on familiar tokens.** Stronger preference for familiar tokens generally improves Pass@1 accuracy while reducing output diversity and limiting subsequent RL improvement.

Across all three approaches, we observe the same pattern: strengthening the preference on the familiar token may improve pre-RL performance, but consistently reduces both the token-level entropy and the answer-category entropy, and also Pass@8 in Appendix D.6. As a result, the model benefits less from subsequent RL, and its initial performance advantage is largely erased by the end of training. The fact that the same phenomenon appears across all three approaches suggests that the diversity collapse is related to their shared preference for more familiar tokens.

5 Understanding Diversity Collapse

We next study why preferring the familiar tokens in SFT can lead to diversity collapse. We let $K = |\mathcal{Y}|$, $\mu \in \Delta^K$ denote the external data distribution and $\pi_t \in \Delta^K$ the model distribution at iteration t. Standard SFT minimizes $\mathcal{L}_{\mathrm{SFT}}(\pi) = -\sum_{i=1}^{K} \mu_i \log \pi_i = H(\mu) + D_{\mathrm{KL}}(\mu \| \pi)$. The unique population optimum of standard SFT is $\pi^\star_{\mathrm{SFT}} = \mu$. Consequently, the optimal solution of standard SFT matches the external data distribution.

A Unified Framework for SFT Methods that Prefer Familiar Tokens. We next consider SFT methods that assign greater weight to tokens that are already familiar to the current model. Let $\mu \in \Delta^K$ denote the external data distribution, and parameterize the model distribution by logits $z_t \in \mathbb{R}^K$: $\pi_t = \mathrm{softmax}(z_t)$. The general population weighted SFT objective at iteration t is

$$\mathcal{L}(z, z_t) = -\sum_{i=1}^{K} \mathrm{sg}(q_{t,i}) \log \pi_i(z), \qquad \pi(z) = \mathrm{softmax}(z),$$

where $\mathrm{sg}(\cdot)$ denotes stop-gradient, q_t denotes the extra weights in SFT determined by the current model distribution π_t. Based on Section 3, we can have $q_{t,i} = \mu_i \pi_{t,i}$ for DFT, $q_{t,i} = \mu_i \mathbf{1}\{\pi_{t,i} \geq \tau\}$ for ProFit, and $q_{t,i} = \mu_i \pi_{t,i}^{1/\delta}$ for TALR.

Diversity Collapse We now study why preferring familiar tokens leads to diversity collapse. Taking DFT $q_{t,i} = \mu_i \pi_{t,i}$ as an example. The gradient evaluated at the current logits is

$$\left.\frac{\partial \mathcal{L}_{\mathrm{DFT}}(z, z_t)}{\partial z_i}\right|_{z=z_t} = \pi_{t,i}\left(\langle \mu, \pi_t \rangle - \mu_i\right).$$

Define the fixed objective $G(z) = -\langle \mu, \mathrm{softmax}(z) \rangle$. Its gradient can be written as

$$\frac{\partial G(z)}{\partial z_i} = \pi_i(z)\left(\langle \mu, \pi(z) \rangle - \mu_i\right).$$

Therefore, $\nabla_z \mathcal{L}_{\mathrm{DFT}}(z_t; z_t) = \nabla_z G(z_t)$, and DFT update is then equivalent to

$$z_{t+1} = z_t - \eta \nabla G(z_t). \tag{3}$$

Thus, the logit update of DFT is equivalent to gradient descent on the fixed objective G, whose minimizer corresponds to a one-hot distribution. Hence, this optimization converges to a one-hot solution and hence leads to diversity collapse.

Theorem 1 (Diversity Collapse under DFT)**.** *Suppose that the entries of μ are pairwise distinct, i.e. $\mu_i \neq \mu_j, \forall i \neq j$. Let $z_0 \in \mathbb{R}^K$, and consider the DFT update*

$$z_{t+1,i} = z_{t,i} + \eta \pi_{t,i}\left(\mu_i - \langle \mu, \pi_t \rangle\right), \qquad \pi_t = \mathrm{softmax}(z_t).$$

If $0 < \eta < 4/5$, then there exists an index $j \in [K]$ such that $\pi_t \to e_j$.

The proof is provided in Appendix B.1. Intuitively, the gradient assigned to token i is proportional to its current probability $\pi_{t,i}$, so tokens that are already familiar to the current model receive stronger updates, while low-probability tokens receive progressively weaker learning signals. This feedback drives the model distribution toward a vertex of the probability simplex and eliminates its diversity. The same perspective also applies to other methods that prefer familiar tokens. Below, we provide an informal statement about theoretical guarantees of other algorithms. The formal statements and proofs are provided in Appendix C.

Theorem 2 (Informal)**.** *Under some regularity conditions, optimization with the TALR with fixed $\delta < 1$ converges to a one-hot vector, while ProFit converges to a k-hot distribution, where k is a constant independent of the vocabulary size.*

6 Base Model Merging

Base-model merging. Section 4 and Section 5 empirically and theoretically show that preferring familiar tokens in SFT, a phenomenon that exists in many previous SFT works, is a key driver of diversity collapse. In this section, we provide a simple yet effective recovery approach: merging the fine-tuned model back to the base model. Let θ_{base} denote the parameters of the pretrained base model and θ_{ft} denote the parameters of the fine-tuned model. In our main experiment, $\theta_{\mathrm{ft}} = \theta_{\mathrm{dft}}$ since DFT is a representative algorithm for preferring the familiar token. We construct a merged checkpoint by linearly interpolating their parameters:

$$\theta_{\mathrm{merge}} = 0.5\theta_{\mathrm{ft}} + 0.5\theta_{\mathrm{base}}.$$

We call this base model merging approach BMM. As we will show later, this merging approach retains the performance gains acquired during fine-tuning while recovering the diversity that is lost by the preference for familiar tokens. Moreover, it is applied without additional training, providing a computationally efficient way to preserve room for subsequent RL improvement. This technique is also mentioned in WiSE-FT (Wortsman et al., 2022) to improve the robustness of fine-tuned model, and some works (Dang et al., 2025; Liu et al., 2026a) also applied it to solve excessive SFT. In contrast, we investigate the effect of BMM in recent SFT variants that prefer familiar tokens. As in Section 6.1, we find that BMM is more effective for SFT variants than for standard SFT such as DFT and ProFit that upweight high-probability tokens, while preserving their accuracy gains.

Baselines. We consider standard SFT and several representative variants, including DFT, TALR, and ProFit. We also compare against two standard regularization approaches for mitigating diversity collapse: KL-divergence regularization ASFT (Zhu et al., 2026), and entropy regularization (DFT-EntReg in Table 1). For KL regularization, we use the forward KL divergence, which has been shown to outperform its reverse KL (Zhu et al., 2026). We further compare with the method that merges the standard SFT checkpoint with the base model (SFT+BMM), to study whether the benefits of BMM are more pronounced for models whose distributions have become overly concentrated.

Table 1: Pass@1 and Pass@8 accuracy (%) on five mathematical reasoning benchmarks. Each entry is reported as Pass@1 / Pass@8. Among models except the base model, the best and second-best results within each model group are shown in bold and underlined, respectively.

Model	Method	AMC23	AIME24	AIME25	MATH-OAI	OlympiadBench	Avg.
Qwen2.5-Math-1.5B	Base	29.2 / 72.7	5.0 / 22.5	2.3 / 11.4	38.4 / 81.4	21.6 / 48.6	19.30 / 47.32
	SFT	30.5 / 64.1	4.2 / 15.9	1.9 / 9.2	57.8 / 82.1	22.5 / 47.6	23.38 / 43.78
	DFT	35.6 / 64.1	7.1 / **16.0**	1.2 / 7.2	63.0 / 79.9	**25.7** / 43.2	26.52 / 42.08
	ASFT	35.9 / 62.0	4.0 / 14.9	3.3 / 10.0	60.6 / 82.3	24.6 / 47.9	25.68 / 43.42
	DFT-EntReg	30.9 / 61.5	**8.5** / 15.9	3.1 / 14.8	58.3 / 80.4	24.6 / 45.3	25.08 / 43.58
	ProFit	33.9 / 64.9	6.2 / 11.7	2.7 / 14.0	59.0 / 81.3	24.0 / 46.2	25.16 / 43.62
	TALR	34.8 / 60.2	6.7 / 13.3	4.8 / 17.1	61.0 / 80.8	24.5 / 46.2	26.36 / 43.52
	SFT+BMM	30.2 / 68.1	4.2 / 14.1	2.9 / 13.4	55.0 / 84.0	22.3 / **48.7**	22.92 / 45.66
	DFT+BMM	**39.8 / 69.0**	6.7 / 15.0	**7.1 / 19.2**	**63.5 / 84.4**	25.2 / 46.8	**28.46 / 46.88**
Qwen2.5-3B	Base	22.3 / 62.3	2.1 / 10.8	0.6 / 5.0	42.7 / 77.9	15.6 / 40.6	16.66 / 39.32
	SFT	20.8 / 50.4	1.5 / 6.2	0.4 / 3.3	47.6 / 74.2	16.0 / 40.2	17.26 / 34.86
	DFT	21.9 / 43.5	1.5 / 8.9	0.6 / 5.0	48.9 / 65.0	16.5 / 33.3	17.88 / 31.14
	ASFT	21.6 / 52.9	1.7 / 9.8	1.0 / 8.3	49.3 / 73.8	15.6 / 35.7	17.84 / 36.10
	DFT-EntReg	**26.9** / 56.9	2.3 / 9.2	0.2 / 1.7	48.6 / 70.6	17.2 / 34.8	19.04 / 34.64
	ProFit	21.1 / 54.3	2.7 / 9.9	1.2 / 10.0	49.8 / 73.5	17.0 / 36.4	18.36 / 36.82
	TALR	24.1 / 52.8	1.5 / 8.2	**2.5 / 14.4**	50.3 / 72.5	16.5 / 35.5	18.98 / 36.68
	SFT+BMM	23.6 / **61.9**	1.9 / 11.4	1.5 / 7.4	48.1 / **77.4**	16.5 / 40.2	18.32 / **39.66**
	DFT+BMM	26.7 / 59.2	**3.1 / 12.5**	0.8 / 6.7	**52.4** / 74.2	**20.9 / 40.8**	**20.78** / 38.68
Qwen3-1.7B-Base	Base	15.9 / 55.9	1.2 / 5.0	1.2 / 8.4	33.1 / 73.8	11.4 / 36.3	12.56 / 35.88
	SFT	23.3 / 57.0	1.5 / 7.4	0.4 / 2.6	48.7 / 75.5	16.2 / 40.7	18.02 / 36.64
	DFT	26.1 / 51.5	1.9 / 3.3	0.8 / 4.7	53.1 / 72.6	18.5 / 36.2	20.08 / 33.66
	ASFT	29.7 / **61.2**	2.9 / 11.3	1.0 / 6.8	54.2 / 78.7	19.9 / 42.5	21.54 / 40.10
	DFT-EntReg	28.6 / 58.3	2.1 / 9.6	0.8 / 5.9	54.2 / 76.7	19.5 / 38.7	21.04 / 37.84
	ProFit	23.6 / 55.0	2.1 / 11.2	0.8 / 4.7	52.4 / 76.3	19.3 / 38.7	19.64 / 37.18
	TALR	28.0 / 56.2	1.5 / 8.9	1.7 / 6.3	50.2 / 74.8	18.8 / 38.6	20.04 / 36.96
	SFT+BMM	25.0 / 58.1	2.9 / **14.3**	**2.1 / 9.2**	47.4 / 78.8	17.5 / 43.5	18.98 / **40.78**
	DFT+BMM	**32.7** / 59.9	**3.3** / 9.1	1.5 / 7.7	**57.8 / 79.9**	**22.5 / 43.7**	**23.56** / 40.06
Qwen3-4B-Base	Base	32.8 / 69.1	7.9 / 22.2	6.9 / 22.2	45.2 / 85.9	21.9 / 51.7	22.94 / 50.22
	SFT	32.2 / 68.4	3.5 / 12.5	4.6 / 17.6	59.6 / 82.2	26.2 / 50.4	25.22 / 46.22
	DFT	33.1 / 57.8	6.7 / 14.2	1.7 / 9.8	63.7 / 76.8	28.3 / 43.5	26.70 / 40.42
	ASFT	35.9 / 71.9	7.7 / 18.4	4.4 / 12.9	64.7 / 82.5	30.1 / 49.7	28.56 / 47.08
	DFT-EntReg	35.0 / 65.0	6.0 / 12.6	4.6 / 15.7	62.9 / 81.4	27.9 / 46.5	27.28 / 44.24
	ProFit	34.4 / 68.5	6.5 / 16.5	2.5 / 11.8	62.8 / 82.2	27.9 / 48.3	26.82 / 45.46
	TALR	38.0 / 67.6	5.0 / 16.4	3.8 / 13.1	60.8 / 79.5	26.5 / 46.4	26.82 / 44.60
	SFT+BMM	35.9 / 76.1	6.9 / **19.9**	2.7 / 14.0	58.9 / **86.3**	26.5 / **53.1**	26.18 / 49.88
	DFT+BMM	**45.8 / 79.5**	**9.4** / 19.3	**6.7 / 22.7**	**69.7** / 85.5	**32.8** / 50.6	**32.88 / 51.52**

6.1 Main Results

Math Reasoning. For math reasoning, we use four representative backbones, Qwen2.5-Math-1.5B, Qwen2.5-3B, Qwen3-1.7B-Base and Qwen3-4B-Base. We use a 10K-example subset of NuminaMath-CoT (Li et al., 2024) as the training dataset, and use the default chat template and chain-of-thought prompts. For the KL and entropy regularization, we set the regularization coefficient to 0.05. We evaluate the resulting models on five math benchmarks AMC 2023, AIME 2024, AIME 2025, MATH-OAI (Hendrycks et al., 2021) and OlympiadBench (He et al., 2024). For each problem, we generate 16 responses at temperature 0.7 with a maximum completion length of 4096 tokens. Then, let c denote the number of correct responses, we compute Pass@1 as $c/16$, and $\text{Pass@8} = 1 - \binom{16-c}{8} / \binom{16}{8}$, which gives the probability that at least one correct response appears among 8 samples.

Table 1 shows the Pass@1 and Pass@8 performance of DFT+BMM and all other baselines. As shown in the table, across all four models, DFT+BMM achieves the highest Pass@1 performance. For example, on Qwen3-1.7B-Base and Qwen3-4B-Base, DFT+BMM outperforms the strongest baseline by an average of 2.02 and 4.32 points respectively across the five benchmarks. These results suggest that base model merging can preserve and even improve the strong Pass@1 performance achieved by DFT by combining its Pass@1 gains with the basic capabilities of the base model. Moreover, DFT+BMM significantly increases Pass@8 relative to DFT and achieves competitive

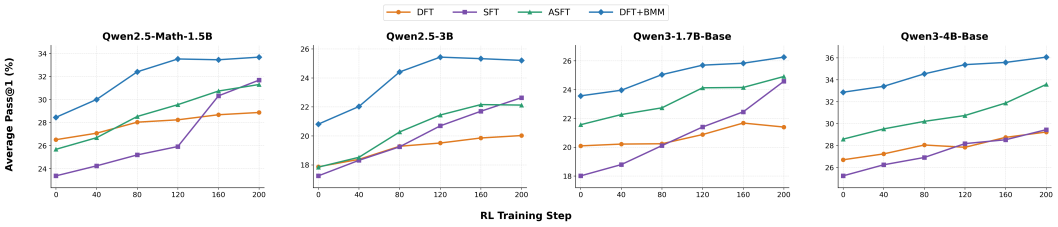


Figure 3: RL performance of DFT+BMM and representative baselines across four models. Performance is averaged over AIME24, AIME25, AMC23, MATH-OAI, and OlympiadBench.

Table 3: Pass@1 accuracy (%) of two models before and after GRPO training on code-generation benchmarks. Bold and underlined values denote the best and second-best results respectively.

(a) Before RL

Method	HE	HE+	MBPP	MBPP+	Avg.
Qwen2.5-Coder-1.5B					
SFT	58.88	53.05	71.48	61.95	61.34
DFT	62.73	56.48	73.33	62.63	63.79
ASFT	63.03	55.79	72.87	62.96	63.66
SFT + BMM	62.27	55.68	72.45	62.60	63.25
DFT + BMM	**66.77**	**59.68**	**75.60**	**65.31**	**66.84**
Qwen2.5-Coder-3B					
SFT	70.96	64.63	75.26	65.66	69.13
DFT	70.39	63.45	76.65	66.01	69.13
ASFT	71.91	65.09	77.20	66.34	70.13
SFT + BMM	71.61	65.59	75.28	65.26	69.43
DFT + BMM	**74.77**	**68.52**	**79.53**	**68.42**	**72.81**

(b) After RL

Method	HE	HE+	MBPP	MBPP+	Avg.
Qwen2.5-Coder-1.5B					
SFT + GRPO	66.69	58.73	74.12	63.49	65.76
DFT + GRPO	65.70	58.80	74.69	63.33	65.63
ASFT + GRPO	67.80	59.87	75.28	64.29	66.81
SFT + BMM + GRPO	68.22	61.66	75.40	64.77	67.51
DFT + BMM + GRPO	**70.66**	**64.37**	**77.43**	**65.71**	**69.54**
Qwen2.5-Coder-3B					
SFT + GRPO	74.24	67.53	77.63	67.18	71.64
DFT + GRPO	73.21	65.74	78.04	66.85	70.96
ASFT + GRPO	74.58	67.07	78.90	66.48	71.76
SFT + BMM + GRPO	74.66	68.37	78.87	**68.40**	72.57
DFT + BMM + GRPO	**76.71**	**71.00**	**80.39**	68.39	**74.12**

Pass@8 performance compared with standard SFT and the base model, suggesting that base-model merging mitigates the over-concentration induced by DFT.

Base Model Merging is More Effective for DFT than SFT. We further compare the gains from merging the base model into standard SFT and DFT checkpoints. For algorithm A, we define $\Delta_{BMM}(\mathrm{A})$ as the performance change induced by base-model merging, i.e., the performance gap between A+BMM and A. As shown in Table 2, base-model merging provides larger improvements when applied to DFT across all four backbones for both Pass@1 and Pass@8. On average, merging improves DFT by 3.63/7.46 points in Pass@1/Pass@8, compared with 0.63/3.62 points for standard SFT. This consistently larger gain on DFT shows that familiarity-biased fine-tuning induces stronger over-concentration, and therefore leaves greater room for base-model merging to recover the suppressed capabilities and diversity.

Table 2: Gains from base-model merging.

Backbone	$\Delta_{\mathrm{BMM}}(\mathrm{SFT})$ P@1	$\Delta_{\mathrm{BMM}}(\mathrm{SFT})$ P@8	$\Delta_{\mathrm{BMM}}(\mathrm{DFT})$ P@1	$\Delta_{\mathrm{BMM}}(\mathrm{DFT})$ P@8
Qwen2.5-Math-1.5B	-0.46	+1.88	**+1.94**	**+4.80**
Qwen2.5-3B	+1.06	+4.80	**+2.90**	**+7.54**
Qwen3-1.7B-Base	+0.96	+4.14	**+3.48**	**+6.40**
Qwen3-4B-Base	+0.96	+3.66	**+6.18**	**+11.10**
Average	+0.63	+3.62	**+3.63**	**+7.46**

Better Initialization for RL. Besides Pass@1 and Pass@8, we further study whether the checkpoints lead to better RL performance. We use the GRPO algorithm (Shao et al., 2024) on the training set with learning rate 10^{-6} and train for 200 steps. Figure 3 compares the RL learning curves of DFT+BMM with representative baselines SFT, DFT, and ASFT. Although DFT begins with relatively strong performance due to its preference for familiar tokens, its reduced diversity leaves limited room for improvement during RL, ultimately resulting in worse final performance than standard SFT. ASFT mitigates this issue through KL regularization and achieves larger RL gains. Notably, DFT+BMM combines DFT's strong initial performance with better diversity from the base model. As shown in Figure 3, compared to DFT, DFT+BMM not only provides a stronger initialization but also achieves larger subsequent RL gain. As a result, it achieves the strongest subsequent RL performance. The results of the final checkpoint on each benchmark are provided in Appendix D.5.

Code Generation. For code generation, we use Qwen2.5-Coder-1.5B and Qwen2.5-Coder-3B as our base models. We use 10K samples of OpenCodeInstruct (Ahmad et al., 2025) as the SFT

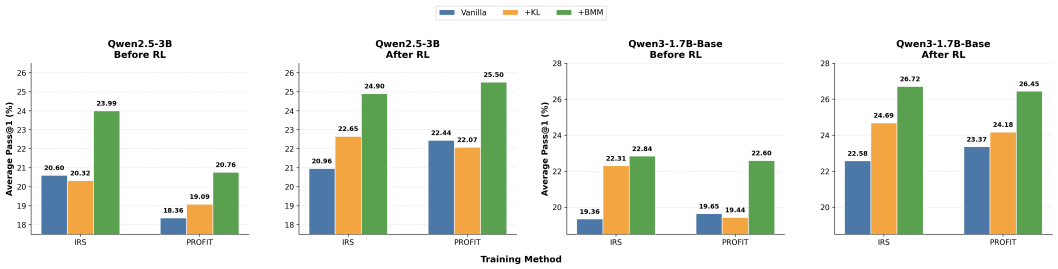


Figure 5: Pass@1 performance of ProFit and IRS before and after RL under three variants: the original method, KL regularization, and BMM.

training dataset. The KL regularization coefficient is 0.05. We evaluate the resulting models on HumanEval (Chen et al., 2021), HumanEval+ (Liu et al., 2023), MBPP (Austin et al., 2021), and MBPP+ (Liu et al., 2023). We compute Pass@1 and Pass@8 as same as math reasoning task. For subsequent RL, we use GRPO and train for 400 steps with learning rate 1×10^{-6} on KodCode-Light-RL-10K (Xu et al., 2025). The experiment details are provided in Appendix E.

BMM Improves Diversity over DFT. We examine whether BMM alleviates the diversity collapse induced by DFT using answer category entropy and rollout token entropy as two metrics. For each model, we compute the paired difference between DFT+BMM and DFT on each prompt. We estimate uncertainty using paired bootstrap resampling over all prompts across the five math benchmarks. As shown in Figure 4, DFT+BMM consistently increases both answer-category entropy and rollout token entropy across all four models. Moreover, all paired 95% confidence intervals lie above zero. These results show that BMM yields statistically significant increases in both entropy metrics.

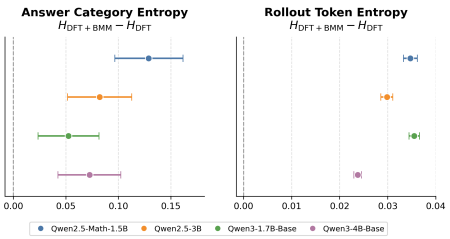


Figure 4: Diversity gain from BMM. Points show the averaged entropy difference $H_{\mathrm{DFT+BMM}} - H_{\mathrm{DFT}}$ over all examples from five benchmarks. Bars denote 95% paired bootstrap confidence intervals.

Model Merging Works Well across Different Methods. We show that BMM works well for other variants that prefer familiar tokens in SFT. We consider ProFit with threshold $\tau = 0.1$ and IRS with rollout interval $N = 500$, and compare the original method with BMM and KL regularization. Figure 5 show that BMM outperforms other two baselines.

In Appendix D, we provide additional ablations on the KL regularization strength and learning rate. We also compare DFT+BMM against DFT with early stopping, to examine whether the gains from BMM can be explained simply by avoiding excessive fine-tuning. In addition, we study the sensitivity of the model-merging coefficient. We also compare merging DFT with the base model against merging DFT with an SFT checkpoint, to investigate whether the SFT checkpoint can provide a better merge partner and further improve performance.

7 Conclusion

We study the effect of preferring familiar tokens in supervised fine-tuning, which is very common in recent SFT variants (Wu et al., 2026; Liu et al., 2026b; Lin et al., 2026). Even though they improve immediate performance, we find that this preference can significantly lead to diversity collapse and limit further improvement through RL. Our theoretical analysis also shows that this preference progressively concentrates the learned policy. To mitigate this problem, we propose merging the fine-tuned model with its pretrained base model. Comprehensive experiments show that base-model merging preserves or improves Pass@1 performance while improves diversity over the original method and enabling stronger subsequent RL performance. More broadly, our results suggest that diversity collapse should be regarded as a key diagnostic when evaluating improvements of SFT, ensuring that improvements in immediate performance are not achieved by over-concentrating.

AI USE STATEMENT

In this work, we used generative AI tools for checking the grammar, help write some supporting code (with human), and generating feedback about the manuscript. We have not used generative AI tools for ideation, proposing research method or designing the main experiments. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We have made every effort to ensure that the results presented in this paper are reproducible. Relative codes are provided in the supplementary material. The experiment setup and training configurations are also provided in Appendix E.

REFERENCES

Ahmad, W. U., Ficek, A., Samadi, M., Huang, J., Noroozi, V., Majumdar, S., and Ginsburg, B. (2025). Opencodeinstruct: A large-scale instruction tuning dataset for code llms. *arXiv preprint arXiv:2504.04030*.

Austin, J., Odena, A., Nye, M., Bosma, M., Michalewski, H., Dohan, D., Jiang, E., Cai, C., Terry, M., Le, Q., et al. (2021). Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*.

Chen, H., Razin, N., Narasimhan, K., and Chen, D. (2025). Retaining by doing: The role of on-policy data in mitigating forgetting. *arXiv preprint arXiv:2510.18874*.

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. D. O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., et al. (2021). Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*.

Cui, G., Zhang, Y., Chen, J., Yuan, L., Wang, Z., Zuo, Y., Li, H., Fan, Y., Chen, H., Chen, W., et al. (2025). The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*.

Dang, X., Baek, C., Wen, K., Kolter, Z., and Raghunathan, A. (2025). Weight ensembling improves reasoning in language models. *arXiv preprint arXiv:2504.10478*.

Diao, M., Yang, L., Gong, W., Zhang, Y., Yan, Z., Han, Y., Liang, K., Xu, W., and Ma, Z. (2026). Entropy-adaptive fine-tuning: Resolving confident conflicts to mitigate forgetting. *arXiv preprint arXiv:2601.02151*.

Dong, H., Xiong, W., Goyal, D., Zhang, Y., Chow, W., Pan, R., Diao, S., Zhang, J., Shum, K., and Zhang, T. (2023). Raft: Reward ranked finetuning for generative foundation model alignment. *arXiv preprint arXiv:2304.06767*.

Hao, Z., Wang, H., Liu, H., Luo, J., Yu, J., Dong, H., Lin, Q., Wang, C., and Chen, J. (2026). Rethinking entropy interventions in rlvr: An entropy change perspective. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 31105–31133.

He, C., Luo, R., Bai, Y., Hu, S., Thai, Z., Shen, J., Hu, J., Han, X., Huang, Y., Zhang, Y., et al. (2024). Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3828–3850.

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. (2021). Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Jiang, Y., Li, Y., Chen, G., Liu, D., Cheng, Y., and Shao, J. (2025). Rethinking entropy regularization in large reasoning models. *arXiv preprint arXiv:2509.25133*.

Kotha, S. and Liang, P. (2026). Replaying pre-training data improves fine-tuning. *CoRR*, abs/2603.04964.

Li, J., Beeching, E., Tunstall, L., Lipkin, B., Soletskyi, R., Huang, S., Rasul, K., Yu, L., Jiang, A. Q., Shen, Z., et al. (2024). Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13(9):9.

Li, L., Zhou, Z., Hao, J., Liu, J., Miao, Y., Pang, W., Tan, X., Chu, W., Wang, Z., Pan, S., et al. (2026). The choice of divergence: A neglected key to mitigating diversity collapse in reinforcement learning with verifiable reward. In *International Conference on Learning Representations*, volume 2026, pages 64742–64768.

Lin, J., Wang, Z., Qian, K., Wang, T., Srinivasan, A., Zeng, H., Jiao, R., Zhou, X., Gesi, J., Wang, D., et al. (2026). Sft doesn't always hurt general capabilities: Revisiting domain-specific fine-tuning in llms. In *International Conference on Learning Representations*, volume 2026, pages 46954–46992.

Liu, J., Xia, C. S., Wang, Y., and Zhang, L. (2023). Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in neural information processing systems*, 36:21558–21572.

Liu, R., Liu, J., Wan, X., Fu, Y., and Pan, L. (2026a). When rl fails after sft: Rejuvenating model plasticity for robust sft-to-rl handoff. *arXiv preprint arXiv:2606.09932*.

Liu, T., Wu, T., Yang, R., Sun, S., Wang, J., and Yang, Y. (2026b). Profit: Leveraging high-value signals in sft via probability-guided token selection. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 15383–15401.

Peng, R., Ren, Y., Yu, Z., Liu, W., and Wen, Y. (2025). Simko: Simple pass@k policy optimization. *CoRR*, abs/2510.14807.

Qin, C. and Springenberg, J. T. (2025). Supervised fine tuning on curated data is reinforcement learning (and can be improved). *arXiv preprint arXiv:2507.12856*.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y., Wu, Y., et al. (2024). Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Tajwar, F., Zeng, G., Zhou, Y., Song, Y., Arora, D., Jiang, Y., Schneider, J., Salakhutdinov, R., Feng, H., and Zanette, A. (2026). Maximum likelihood reinforcement learning. *arXiv preprint arXiv:2602.02710*.

Wang, K., Li, S., Salzmann, M., and Frossard, P. (2026). Prift: Prior-support guided supervised fine-tuning. *arXiv preprint arXiv:2606.09396*.

Wortsman, M., Ilharco, G., Kim, J. W., Li, M., Kornblith, S., Roelofs, R., Lopes, R. G., Hajishirzi, H., Farhadi, A., Namkoong, H., et al. (2022). Robust fine-tuning of zero-shot models. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7949–7961. IEEE.

Wu, T., Li, X., and Liu, P. (2025). Progress or regress? self-improvement reversal in post-training. In *International Conference on Learning Representations*, volume 2025, pages 11959–11979.

Wu, Y., Zhou, Y., Zhou, Z., Peng, Y., Ye, X., Hu, X., Zhu, W., Qi, L., Yang, M.-H., et al. (2026). On the generalization of sft: A reinforcement learning perspective with reward rectification. In *International Conference on Learning Representations*, volume 2026, pages 27550–27571.

Xiong, W., Yao, J., Xu, Y., Pang, B., Wang, L., Sahoo, D., Li, J., Jiang, N., Zhang, T., Xiong, C., et al. (2025). A minimalist approach to llm reasoning: from rejection sampling to reinforce. *arXiv preprint arXiv:2504.11343*.

Xu, Z., Liu, Y., Yin, Y., Zhou, M., and Poovendran, R. (2025). Kodcode: A diverse, challenging, and verifiable synthetic dataset for coding. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 6980–7008.

Yu, Q., Zhang, Z., Zhu, R., Yuan, Y., Zuo, X., Yue, Y., Dai, W., Fan, T., Liu, G., Liu, L., et al. (2026). Dapo: An open-source llm reinforcement learning system at scale. *Advances in Neural Information Processing Systems*, 38:113222–113244.

Yuan, X., Zhang, C., Liu, Z., Shi, D., Pan, L., Vosoughi, S., and Lee, W. (2025). Superficial self-improved reasoners benefit from model merging. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 5912–5932.

Zhang, D., Dai, Q., and Peng, H. (2026a). The best instruction-tuning data are those that fit. *Advances in Neural Information Processing Systems*, 38:141172–141208.

Zhang, D., Xu, Y., Wang, H., Chen, Q., and Peng, H. (2026b). Good sft optimizes for sft, better sft prepares for reinforcement learning. *arXiv preprint arXiv:2602.01058*.

Zhang, M., Liu, Y., Lin, S., Yang, X., Dai, Q., Luo, C., Jiang, W., Hou, P., Zeng, A., Geng, X., et al. (2026c). Towards on-policy sft: Distribution discriminant theory and its applications in llm training. *arXiv preprint arXiv:2602.12222*.

Zhu, H., Su, J., Lai, P., Ma, R., Zhang, W., Yang, L., and Chen, G. (2026). Anchored supervised fine-tuning. In *International Conference on Learning Representations*, volume 2026, pages 110364–110385.

A Related Works and Discussions

A.1 Detailed Discussion of SFT Variants

Token-level and trajectory-level reweighting. A recent line of work modifies standard SFT by assigning non-uniform weights to supervised tokens according to the model's predictive distribution. DFT (Wu et al., 2026) derives standard SFT from an RL perspective and observes that its gradient implicitly places disproportionately large updates on low-probability target tokens. To rectify this effect, DFT multiplies each token-level cross-entropy loss by the detached probability of the target token under the current model, thereby placing relatively more emphasis on tokens that are already supported by the model. ProFit (Liu et al., 2026b) arrives at a related design from the perspective of the one-to-many nature of language: it argues that high-probability tokens are more likely to capture the core reasoning structure, whereas many low-probability tokens correspond to replaceable surface expressions, and therefore selectively masks low-probability tokens during SFT. TALR (Lin et al., 2026) similarly downweights difficult tokens, but derives its weighting rule through a constrained optimization formulation. Its token weights are approximately proportional to $\pi_\theta(y_t \mid x, y_{<t})^{1/\delta}$, together with a lower-bound constraint to prevent very hard tokens from receiving vanishing weights.

These methods compute token weights from the model being optimized, causing the weighting signal itself to evolve during training. PriFT (Wang et al., 2026) highlights that such online reweighting can introduce a self-reinforcing dynamic, since familiar tokens receive increasingly larger weights. Instead, PriFT computes token-level support using a frozen pretrained model and uses this fixed distribution $\omega_\theta = \pi_{\text{base}}(y_t \mid x, y_{<t})$. ASFT (Zhu et al., 2026) addresses a related issue from a regularization perspective. Building on DFT, it observes that probability-based online reweighting can induce progressive distributional drift and augments the objective with a forward-KL anchoring term to keep the fine-tuned policy close to a reference distribution. EAFT (Diao et al., 2026) considers not only the probability of the token but also the uncertainty of the token. It identifies low-probability but low-entropy tokens as "confident conflicts" and uses token entropy as a gating signal to suppress updates on such conflicts.

There are also some works study whether we can assign trajectory-level weights or use different data resources for each trajectory in the data pool. iw-SFT (Qin and Springenberg, 2025) derives an importance-weighted variant of SFT that reweight the training trajectories. GRAPE (Zhang et al., 2026a) considers settings where multiple candidate responses are available for each instruction and selects the response assigned the highest likelihood by the target model before performing standard SFT. (Kotha and Liang, 2026) replays the data in the pre-training phase and show that the replay data can help improve the model's performance.

Self-generated and on-policy training. A more direct way to align the training distribution with the model and training more on familiar tokens is to generate training data from the model itself. RAFT (Dong et al., 2023) iteratively samples multiple responses from the current model, ranks or filters them using a reward signal, and fine-tunes the model on the selected responses before generating a new batch with the updated policy. Consequently, the training data is periodically refreshed as the policy changes. Similarly, Chen et al. (2025) study the role of on-policy, i.e., self-generated data in catastrophic forgetting and introduce Iterative-SFT, which repeatedly regenerates training responses from the latest model. They find that such approximately on-policy SFT substantially reduces forgetting compared with SFT on fixed external demonstrations.

B Proofs for Section 5

B.1 Proof of Theorem 1

Recall the fixed objective $G(z) = -\langle \mu, \mathrm{softmax}(z) \rangle$. Let $\pi(z) = \mathrm{softmax}(z)$. The gradient of G can be written as

$$\frac{\partial G(z)}{\partial z_i} = -\pi_i(z)\big(\mu_i + G(z)\big).$$

Therefore, by Eq. 3, the DFT update can be written as

$$z_{t+1} = z_t - \eta \nabla G(z_t).$$

We first establish that G is globally smooth. Let $g(z) = -\nabla G(z)$. A direct calculation gives

$$\nabla^2 G(z) = -\operatorname{diag}(g(z)) + g(z)\pi(z)^\top + \pi(z)g(z)^\top.$$

Since $\mu \in \Delta^K$,

$$|g_i(z)| = \pi_i(z)|\mu_i + G(z)| \le \pi_i(z)(1 - \pi_i(z)) \le \frac{1}{4}.$$

Moreover, $\|g(z)\|_2 \le \|g(z)\|_1 \le 1$, and $\|\pi(z)\|_2 \le 1$. It follows that

$$\|\nabla^2 G(z)\|_{\mathrm{op}} \le \frac{1}{4} + 2 < \frac{5}{2}.$$

Thus, the function G is L-smooth with $L = 5/2$. Now for $0 < \eta < 2/L = 4/5$, the standard descent lemma yields

$$G(z_{t+1}) \le G(z_t) - \eta\left(1 - \frac{L\eta}{2}\right)\|\nabla G(z_t)\|_2^2.$$

Because G is bounded below, summing this inequality over t gives

$$\sum_{t=0}^{\infty} \|\nabla G(z_t)\|_2^2 < \infty,$$

and hence

$$\|\nabla G(z_t)\|_2 \to 0.$$

Since the sequence $\{\pi_t\}$ is bounded, it has at least one convergence point. Let $\bar{p}$ be any such convergence point. By continuity of the gradient,

$$\bar{\pi}_i\left(\mu_i - \langle \mu, \bar{p}\rangle\right) = 0, \qquad \forall i \in [K].$$

Consequently, every index in the support of $\bar{p}$ must satisfy

$$\mu_i = \langle \mu, \bar{p}\rangle.$$

Since the entries of μ are pairwise distinct, the support of $\bar{p}$ contains only one index. Therefore, every convergence point of $\{\pi_t\}$ is a vertex of the probability simplex: $\bar{p} \in \{e_1, \ldots, e_K\}$.

Finally, since $\|z_{t+1} - z_t\|_2 = \eta\|\nabla G(z_t)\|_2 \to 0$ and softmax is Lipschitz, we also have

$$\|\pi_{t+1} - \pi_t\|_2 \to 0.$$

Hence, the set of convergence points of a bounded sequence is connected. Because the convergence points only contain one-hot vector which is disconnected, this set must only contain a single point. Hence, there exists $j \in [K]$ such that

$$\pi_t \to e_j.$$

□

C Extensions to Other Methods that Prefer the Familiar

We next analyze TALR (Lin et al., 2026) and ProFit (Liu et al., 2026b). These methods implement the preference for familiar outputs in different ways: TALR continuously reweights token probabilities, and ProFit removes low-probability tokens from the training data. Recall that

$$\mathcal{L}_t(z) = -\sum_{i=1}^{K} \operatorname{sg}(q_{t,i}) \log \pi_i(z), \qquad \pi(z) = \operatorname{softmax}(z),$$

and $q_{t,i}$ for different approaches can be written as

$$q_{t,i} = \begin{cases} \mu_i \pi_{t,i}^{1/\delta}, & \text{TALR} \\ \mu_i \mathbf{1}\{\pi_{t,i} \ge \tau\}, & \text{ProFit} \end{cases}$$

C.1 TALR

In this subsection, we analyze TALR algorithm. In fact, the TALR algorithm uses batch loss to dynamic adjust τ and also adds a floor gate parameter $\omega_{\min}$ in practice. For the theoretical simplicity, we only consider $q_{t,i} = \mu_i \pi_{t,i}^{1/\delta}$. Let $\gamma = \frac{1}{\delta}$. When $\delta < 1$, we have $\gamma > 1$. The weighted SFT loss at iteration t is

$$\mathcal{L}_t^{\mathrm{TALR}}(z) = -\sum_{i=1}^{K} \mu_i \, \mathrm{sg}\big(\pi_{t,i}^{\gamma}\big) \log \pi_i(z).$$

Its gradient at $z = z_t$ is

$$\left.\frac{\partial \mathcal{L}_t^{\mathrm{TALR}}(z)}{\partial z_i}\right|_{z=z_t} = \left(\sum_{j=1}^{K} \mu_j \pi_{t,j}^{\gamma}\right) \pi_{t,i} - \mu_i \pi_{t,i}^{\gamma}.$$

The corresponding logit update is therefore

$$z_{t+1,i} = z_{t,i} + \eta \left(\mu_i \pi_{t,i}^{\gamma} - \left(\sum_{j=1}^{K} \mu_j \pi_{t,j}^{\gamma}\right) \pi_{t,i} \right).$$

Define the objective $G_\gamma(z) = -\frac{1}{\gamma} \sum_{i=1}^{K} \mu_i \pi_i(z)^\gamma$. Its gradient satisfies

$$\frac{\partial G_\gamma(z)}{\partial z_i} = -\mu_i \pi_i(z)^\gamma + \pi_i(z) \left(\sum_{j=1}^{K} \mu_j \pi_j(z)^\gamma\right).$$

Thus, TALR is equivalent to gradient descent on G_γ:

$$z_{t+1} = z_t - \eta \nabla G_\gamma(z_t).$$

Theorem 3 (Diversity Collapse under TALR)**.** *Suppose that $\mu_i > 0$ for every i and $\mu_i \neq \mu_j$ for any $i, j \in [K]$, $\delta < 1$, and let $\gamma = 1/\delta > 1$. Consider the TALR update with $0 < \eta < \frac{4}{6\gamma - 1}$. If the trajectory does not converge to a unique global maximum of G_γ, then there exists $j \in [K]$ such that*

$$\pi_t \to e_j.$$

Proof. Recall that

$$G_\gamma(z) = -\frac{1}{\gamma} \sum_{i=1}^{K} \mu_i \pi_i(z)^\gamma, \qquad \pi(z) = \mathrm{softmax}(z),$$

and that the TALR update is equivalent to gradient descent:

$$z_{t+1} = z_t - \eta \nabla G_\gamma(z_t).$$

We first derive the smoothness constant for G_γ. Define

$$a_i = \mu_i \pi_i^\gamma, \qquad Q = \sum_{i=1}^{K} a_i, \qquad h_i = a_i - Q\pi_i.$$

A direct calculation gives

$$\nabla^2 G_\gamma(z) = -\gamma \left[\mathrm{diag}(h) - h\pi^\top - \pi h^\top\right] - (\gamma - 1) Q \left[\mathrm{diag}(\pi) - \pi\pi^\top\right].$$

Since $h_i = \pi_i \left(\mu_i \pi_i^{\gamma-1} - Q\right)$ where both $\mu_i \pi_i^{\gamma-1}$ and Q lie in $[0, 1]$, we have

$$|h_i| \leq \pi_i(1 - \pi_i) \leq \frac{1}{4}$$

and

$$\|h\|_2 \leq \|h\|_1 \leq 1.$$

Therefore,

$$\left\|\mathrm{diag}(h) - h\pi^\top - \pi h^\top\right\|_{\mathrm{op}} \le \frac{5}{2}.$$

Moreover, since $Q \le 1$ and $\left\|\mathrm{diag}(\pi) - \pi\pi^\top\right\|_{\mathrm{op}} \le \frac{1}{2}$, we can get

$$\|\nabla^2 G_\gamma(z)\|_{\mathrm{op}} \le \frac{5\gamma}{2} + \frac{\gamma - 1}{2} = 3\gamma - \frac{1}{2}.$$

Thus, ∇G_γ is globally $3\gamma - \frac{1}{2}$-Lipschitz. Then, with

$$0 < \eta < \frac{2}{L_\gamma} = \frac{4}{6\gamma - 1},$$

By the descent lemma,

$$G_\gamma(z_{t+1}) \le G_\gamma(z_t) - \eta\left(1 - \frac{L_\gamma \eta}{2}\right)\|\nabla G_\gamma(z_t)\|_2^2.$$

Since G_γ is bounded below, summing this inequality over t gives

$$\sum_{t=0}^{\infty} \|\nabla G_\gamma(z_t)\|_2^2 < \infty$$

and then $\|\nabla G_\gamma(z_t)\|_2 \to 0$.

We next characterize the stationary points. Let $\bar{p}$ be an convergence point of $\{\pi_t\}$ and define

$$\bar{Q} = \sum_{j=1}^{K} \mu_j \bar{\pi}_j^\gamma.$$

By continuity of the gradient,

$$\bar{\pi}_i\left(\mu_i \bar{\pi}_i^{\gamma-1} - \bar{Q}\right) = 0, \qquad \forall i \in [K].$$

Thus, every index in the support $S = \{i : \bar{\pi}_i > 0\}$ satisfies $\mu_i \bar{\pi}_i^{\gamma-1} = \bar{Q}$. For any fixed support S, the corresponding stationary distribution is therefore

$$\bar{\pi}_i = \frac{\mu_i^{-1/(\gamma-1)}}{\sum_{j\in S} \mu_j^{-1/(\gamma-1)}} \mathbf{1}\{i \in S\}.$$

Since $-(ax^\gamma + by^\gamma)$ is strictly concave for $a > 0$, $b > 0$, and $\gamma > 1$, any stationary distribution whose support satisfies $|S| \ge 2$ is a strict maximizer along some feasible directions and therefore cannot be a local minimum. Hence, the only local minima of G_γ over the simplex are the vertices $e_1, \ldots, e_K$. By assumption, the training process will not be trapped in a local maximum. Hence, every convergence point belongs to the finite set $\{e_1, \ldots, e_K\}$. Then the rest of the proof is the same as Section B.1. □

C.2 ProFit

ProFit uses the hard mask $q_{t,i} = \mu_i \mathbf{1}\{\pi_{t,i} \ge \tau\}$. Define $S_t = \{i \in [K] : \pi_{t,i} \ge \tau\}$. The exact minimizer of the masked cross-entropy loss is

$$\tilde{p}_{t+1,i} = \frac{\mu_i \mathbf{1}\{i \in S_t\}}{\sum_{j\in S_t} \mu_j}.$$

For the theoretical simplicity, we assume that the next-step $\pi_{t+1,i} = \tilde{p}_{t+1,i}$ is the exact minimizer of the loss of the previous step. This is a simplification of the gradient descent dynamics but can help us to illustrate the theoretical insight.

Theorem 4 (Support Collapse under ProFit)**.** *Suppose that $\mu_i > 0$ for every i, $\tau > 0$, and $S_0 \neq \varnothing$. Under the exact masked population update above, there exists a nonempty set $S_\infty \subseteq [K]$ such that*

$$\pi_t \longrightarrow \frac{\mu_i \mathbf{1}\{i \in S_\infty\}}{\sum_{j \in S_\infty} \mu_j}.$$

Moreover,

$$|S_\infty| \leq \left\lfloor \frac{1}{\tau} \right\rfloor.$$

Thus, for a fixed threshold τ, ProFit converges to a k-hot distribution with k independent of the vocabulary size.

Proof. By construction,

$$\pi_{t+1,i} = 0 \qquad \text{for every } i \notin S_t.$$

Consequently,

$$S_{t+1} \subseteq S_t.$$

The active sets therefore form a nested sequence of subsets of $[K]$. Furthermore, every active set satisfies

$$|S_t|\tau \leq \sum_{i \in S_t} \pi_{t,i} \leq 1,$$

and hence

$$|S_t| \leq \left\lfloor \frac{1}{\tau} \right\rfloor.$$

Since S_t is nonempty, the normalized distribution over S_t contains at least one entry no smaller than $1/|S_t| \geq \tau$. Therefore, S_{t+1} is also nonempty.

Because the sequence $S_0 \supseteq S_1 \supseteq \cdots$ is nested and $[K]$ is finite, it stabilizes after finitely many iterations. That is, there exist t_0 and a nonempty set S_∞ such that

$$S_t = S_\infty, \qquad \forall t \geq t_0.$$

Once the active set stabilizes, the update also stabilizes at

$$\pi_{\infty,i} = \frac{\mu_i \mathbf{1}\{i \in S_\infty\}}{\sum_{j \in S_\infty} \mu_j}.$$

The support-size bound follows from the preceding inequality. □

D Additional Experiment Results

D.1 Ablation Study

D.1.1 KL Coefficient

In this subsection, we study the effect of the KL coefficient and see whether base model merging consistently outperform different KL coefficient. As shown in Figure 6, ASFT underperforms base-model merging both before and after RL across all KL coefficients. Small coefficients, such as 0.05 and 0.1, impose weaker constraints and preserve substantially less of the base model's support than model merging. Larger coefficients like 0.5 retain broader support but make training less stable, substantially reducing Pass@1 performance before RL.

D.1.2 Learning Rate

We further investigate whether the benefits of base-model merging remain consistent across different learning rates. Specifically, we train DFT using learning rates of 1×10^{-5}, 2×10^{-5}, and 5×10^{-5}, and merge each resulting checkpoint with its corresponding base model. As shown in Table 4, base-model merging consistently improves the average Pass@1 and Pass@8 performance over the corresponding DFT checkpoint across all learning rates and model scales. These results demonstrate that the effectiveness of base-model merging is robust to the choice of learning rate.

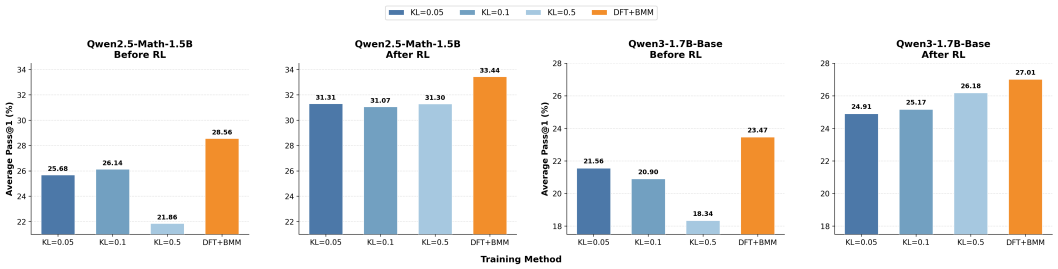


Figure 6: Ablation study of the KL coefficient before and after reinforcement learning. We report the average Pass@1 accuracy across AMC23, AIME24, AIME25, MATH-OAI, and OlympiadBench on Qwen2.5-Math-1.5B and Qwen3-1.7B-Base. Our method consistently achieves the best performance both before and after RL across the two model families.

Table 4: Learning-rate ablation for DFT and DFT+BMM on five mathematical reasoning benchmarks. Each entry reports Pass@1 / Pass@8 accuracy (%). Avg. denotes the macro-average across the five benchmarks.

LR	Method	AMC23	AIME24	AIME25	MATH-OAI	Olympiad	Avg.
Qwen3-1.7B-Base							
1×10^{-5}	DFT	26.1 / 51.5	1.9 / 3.3	0.8 / 4.7	53.1 / 72.6	18.5 / 36.2	20.08 / 33.66
	DFT+BMM	**32.7 / 59.9**	**3.3 / 9.1**	**1.5 / 7.7**	**57.8 / 79.9**	**22.5 / 43.7**	**23.56 / 40.06**
2×10^{-5}	DFT	23.8 / 49.5	1.0 / 5.6	0.6 / 5.0	49.8 / 68.1	16.2 / 32.4	18.28 / 32.12
	DFT+BMM	**30.3 / 59.9**	**2.3 / 12.9**	**1.7 / 9.9**	**55.9 / 77.1**	**20.9 / 42.2**	**22.22 / 40.40**
5×10^{-5}	DFT	14.7 / 33.8	1.5 / 7.7	0.2 / 1.7	34.4 / 52.6	8.6 / 21.1	11.88 / 23.38
	DFT+BMM	**23.9 / 45.2**	**2.5 / 11.3**	**1.5 / 8.2**	**49.3 / 72.8**	**17.9 / 36.9**	**19.02 / 34.88**
Qwen3-4B-Base							
1×10^{-5}	DFT	33.1 / 57.8	6.7 / 14.2	1.7 / 9.8	63.7 / 76.8	28.3 / 43.5	26.70 / 40.42
	DFT+BMM	**45.8 / 79.5**	**9.4 / 19.3**	**6.7 / 22.7**	**69.7 / 85.5**	**32.8 / 50.6**	**32.88 / 51.52**
2×10^{-5}	DFT	30.3 / 51.5	**8.5** / 13.0	2.3 / 8.2	57.7 / 72.7	24.4 / 40.3	24.64 / 37.14
	DFT+BMM	**45.9 / 74.4**	7.9 / **17.4**	**6.7 / 14.9**	**67.2 / 82.8**	**32.1 / 51.6**	**31.96 / 48.22**
5×10^{-5}	DFT	15.0 / 27.1	0.0 / 0.0	0.2 / 1.7	36.1 / 52.4	10.6 / 22.7	12.38 / 20.78
	DFT+BMM	**35.6 / 65.0**	**5.6 / 15.8**	**3.3 / 12.7**	**60.8 / 78.9**	**27.3 / 45.6**	**26.52 / 43.60**

D.2 Comparison with DFT with Early Stopping

One possible interpretation of base-model merging is that it approximates early stopping by moving the fully trained model closer to its initialization. To examine this assumption, we compare DFT+BMM with intermediate DFT checkpoints. As shown in Table 5, DFT+BMM outperforms DFT-Step100, DFT-Step200, and the final DFT model on both metrics.

Table 5: Pass@1 and Pass@8 accuracy (%) comparing DFT+BMM with DFT with learning rate, intermediate and final DFT checkpoints. Each entry is Pass@1 / Pass@8. The best result in each column is highlighted in bold.

Model	Method	AMC23	AIME24	AIME25	MATH-OAI	OlympiadBench	Avg.
Qwen3-1.7B-Base	DFT-Step100	26.3 / 53.3	1.2 / 6.5	1.3 / **8.4**	51.5 / 71.5	16.9 / 33.4	19.43 / 34.62
	DFT-Step200	24.4 / 53.8	2.5 / **13.0**	0.0 / 0.0	50.3 / 71.4	17.8 / 35.0	18.99 / 34.64
	DFT-Final	26.1 / 51.5	1.9 / 3.3	0.8 / 4.7	53.1 / 72.6	18.5 / 36.2	20.09 / 33.66
	DFT+BMM	**32.7 / 59.9**	**3.3** / 9.1	**1.5** / 7.7	**57.8 / 79.9**	**22.5 / 43.7**	**23.56 / 40.06**

Hence, early stopping and DFT+BMM produce fundamentally different models. Early stopping selects a model directly from the optimization trajectory, truncating both the learning of useful task directions and the magnitude of the parameter update. In contrast, DFT+BMM first obtains the complete DFT update and then interpolates the DFT model with the base model. This allows it to

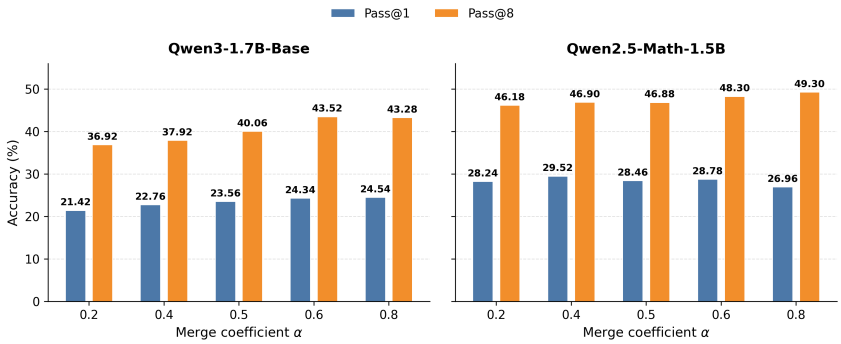


Figure 7: The Pass@1 and Pass@8 accuracy of DFT + BMM with different merging coefficient.

Table 6: Pass@1 and Pass@8 accuracy (%) comparing DFT+SFT and DFT+BMM model merging. Each entry is Pass@1 / Pass@8.

Model	Method	AMC23	AIME24	AIME25	MATH-OAI	OlympiadBench	Avg.
Qwen2.5-Math-1.5B	DFT+SFT	35.2 / 66.9	3.5 / **15.4**	4.2 / 15.7	58.8 / 83.5	23.0 / **47.7**	24.94 / 45.84
	DFT+BMM	**39.8 / 69.0**	**6.7** / 15.0	**7.1 / 19.2**	**63.5 / 84.4**	**25.2** / 46.8	**28.46 / 46.88**
Qwen3-1.7B-Base	DFT+SFT	27.2 / 58.9	2.7 / **11.5**	**1.9 / 11.5**	53.2 / 77.3	20.2 / 41.9	21.03 / **40.22**
	DFT+BMM	**32.7 / 59.9**	**3.3** / 9.1	1.5 / 7.7	**57.8 / 79.9**	**22.5 / 43.7**	**23.56** / 40.06

retain useful directions learned later in training while suppressing excessive deviation from the base model.

D.3 Comparison for Different Merge Coefficients

In this subsection we study whether the base model merging approach is sensitive to the merging coefficient. We choose $\alpha \in \{0.2, 0.4, 0.5, 0.6, 0.8\}$, and merge the model by

$$\theta_{\text{final}} = \alpha\theta_{\text{base}} + (1-\alpha)\theta_{\text{DFT}}.$$

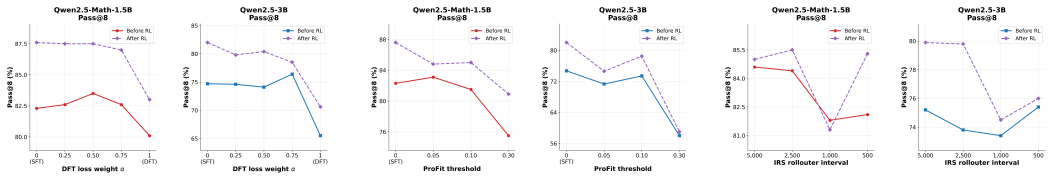
As shown in Figure 7, when α increases, it places more weight on the base model and generally improves Pass@k, indicating better preservation of model diversity. For Pass@1 accuracy, α around 0.5 and 0.6 performs best. Based on these results, we use $\alpha = 0.5$ throughout our experiments as a robust default, and it is robust to most of the scenarios.

D.4 Comparison for Merging with SFT Model

We next investigate what properties make a model a good merging partner for recent variants that over-concentrating the policy. In particular, we compare merging a DFT checkpoint with either its pretrained base model or a standard SFT checkpoint. The SFT model has higher Pass@1 and similar Pass@8 compared with the base model, and therefore serves as a useful test of whether a stronger partner necessarily leads to better merged performance. We find that merging with the stronger SFT checkpoint does not yield a clear advantage over merging with the base model, despite its substantially higher standalone Pass@1. This suggests that a strong merge partner is not determined solely by task accuracy or Pass@k. Retaining complementary support for diversity recovery could be more important, and base model can be a better merge partner. Moreover, merging with SFT model needs training for two times, which is much more computationally-expensive than merging with base model.

D.5 RL Performance by Math Benchmark

In Figure 3, we show the accuracy curve averaged across five math benchmarks AMC23, AIME24, AIME25, MATH-OAI, Olympiadbench. Here, we provide detailed results for the final checkpoint at step 200. As shown in Table 6, DFT + BMM outperforms previous baselines for 4.82-6.92 points



(a) DFT-SFT with varying α. (b) ProFit with varying threshold τ. (c) IRS with varying rollout interval.

Figure 8: Pass@8 before and after RL under varying strengths of preference for familiar tokens.

Table 7: Pass@1 accuracy (%) at checkpoint 200 on five mathematical reasoning benchmarks. The best and second-best results within each model group are shown in bold and underlined, respectively.

Model	Method	AMC23	AIME24	AIME25	MATH-OAI	OlympiadBench	Avg.
Qwen2.5-Math-1.5B	ASFT	45.6	8.5	4.8	67.5	30.1	31.30
	SFT	46.7	6.9	5.4	69.4	30.1	31.70
	DFT	39.1	6.9	4.4	66.5	27.6	28.90
	DFT+BMM	**48.0**	**9.6**	**6.5**	**71.7**	**32.8**	**33.72**
Qwen2.5-3B	ASFT	26.9	4.6	**1.2**	56.5	21.4	22.12
	SFT	29.1	5.0	1.0	57.5	20.6	22.64
	DFT	26.6	2.3	0.4	52.6	18.3	20.04
	DFT+BMM	**34.5**	**5.8**	0.4	**61.4**	**23.8**	**25.18**
Qwen3-1.7B-Base	ASFT	33.6	3.1	**4.0**	60.1	23.8	24.92
	SFT	34.1	4.0	3.8	58.6	22.5	24.60
	DFT	27.0	2.3	0.6	56.4	20.7	21.40
	DFT+BMM	**34.2**	**6.2**	3.1	**62.3**	**25.3**	**26.22**
Qwen3-4B-Base	ASFT	43.6	**10.6**	8.5	71.2	33.9	33.56
	SFT	37.8	6.7	5.2	66.8	30.7	29.44
	DFT	40.8	6.9	3.8	65.5	29.3	29.26
	DFT+BMM	**51.2**	9.2	**9.4**	**74.2**	**36.4**	**36.08**

across four different models, which illustrates the effectiveness of BMM in improving further RL performance.

D.6 Pass@8 Before and After RL with Varying Preference for Familiar Tokens

We provide the Pass@8 results for different degree of preference on familiar tokens. The experiment setup is the same as Section 4. Figure 8 shows the results. Strengthening the preference for high-probability familiar tokens leads to diversity collapse, causing Pass@8 to drop substantially.

D.7 Comparison for PriFT (Wang et al., 2026)

A recent SFT variant PriFT also follows DFT but weights each token by the base model's probability:

$$\mathcal{L}_\omega(\theta) = -\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\sum_{t=1}^{T} \mathrm{sg}(\pi_{\text{base}}(x, y_{\le t}))\log \pi_\theta(y_t \mid x, y_{<t})\right], \tag{4}$$

Table 8 shows the comparison results. On Qwen2.5-math-1.5B and Qwen2.5-3B, DFT + BMM has both higher post-SFT accuracy and RL performance. For Qwen3-1.7B-Base, DFT+BMM has slightly higher post-SFT accuracy with almost same RL performance. Moreover, PriFT needs to additionally load a reference policy during training, which has higher memory cost than DFT+BMM.

E Experiment Details

Table 9 summarizes the main hyperparameters used for training and evaluation.

Table 8: Pass@1 accuracy (%) of PriFT and DFT + BMM before and after GRPO training on math benchmarks.

Method	AMC23	AIME24	AIME25	MATH500	Olym.	Avg.
Qwen2.5-Math-1.5B						
PriFT	38.3	6.0	4.2	61.7	**25.3**	27.10
DFT + BMM	**39.8**	**6.7**	**7.1**	**63.5**	25.2	**28.46**
PriFT + GRPO	45.3	7.3	**6.9**	65.3	28.9	30.74
DFT + BMM + GRPO	**48.0**	**9.6**	6.5	**71.7**	**32.8**	**33.72**
Qwen2.5-3B						
PriFT	21.7	2.5	**0.8**	50.4	17.6	18.60
DFT + BMM	**26.7**	**3.1**	**0.8**	**52.4**	**20.9**	**20.78**
PriFT + GRPO	31.1	5.2	**2.3**	59.1	23.6	24.26
DFT + BMM + GRPO	**34.5**	**5.8**	0.4	**61.4**	**23.8**	**25.18**
Qwen3-1.7B-Base						
PriFT	30.5	2.5	**2.7**	56.0	21.1	22.56
DFT + BMM	**32.7**	**3.3**	1.5	**57.8**	**22.5**	**23.56**
PriFT + GRPO	**35.0**	**6.5**	**3.5**	61.8	24.5	**26.26**
DFT + BMM + GRPO	34.2	6.2	3.1	**62.3**	**25.3**	26.22

Table 9: Hyperparameters for math reasoning and code generation experiments.

Hyperparameter	**Math Reasoning**	**Code Generation**
Indirect on-policy training (SFT, DFT, ProFit, TALR)		
Effective batch size	32	32
Learning rate	1×10^{-5}	1×10^{-5}
Training data size	10000	10000
Maximum sequence length	4096	2048
LR scheduler	Linear	Cosine
Warmup ratio	0	0.05
Precision	FP16	FP16
IRS		
Effective batch size	32	32
Learning rate	1×10^{-5}	1×10^{-5}
Training data size	5000	5000
Chunk size	500 prompts	500 prompts
Rollouts per prompt	8	8
Maximum completion length	2048	2048
Rollout temperature	0.7	0.7
LR scheduler	Constant	Constant
Precision	BF16	BF16
Reinforcement learning		
Effective batch size	64	64
Learning rate	1×10^{-6}	1×10^{-6}
Training steps	200	400
Rollouts per prompt	8	8
Maximum completion length	2048	2048
Mask truncated responses	`True`	`True`
Rollout temperature	1.0	1.0
LR scheduler	Cosine decay	Cosine decay
Warmup	10 steps	10 steps
Precision	BF16	BF16
Evaluation		
Number of generations	16 per problem	16 per problem
Maximum completion length	4096	2048
Sampling temperature	0.7	0.7
Precision	BF16	BF16