BALANCING SPECIALISTS WITH SAFETY: LAGRANGIAN ANCHORED MULTI-TEACHER ON-POLICY DISTILLATION FOR LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-Teacher On-Policy Distillation (MOPD) merges diverse expert capabilities into a single student model. However, current methods rely on rigid routing or heuristics, making them highly vulnerable to multi-domain gradient conflicts. These unconstrained updates often cause a severe "see-saw" effect, where acquiring new skills catastrophically erodes previously established foundational capabilities. To address this, we propose Lagrangian Anchored Multi-Teacher Post-Training (**LAMP**), a framework that protects foundational knowledge by dynamically constraining policy updates. LAMP anchors the student to an initial reference model and employs a self-adjusting penalty mechanism: when aggressive specialist updates threaten to distort core capabilities, LAMP automatically scales up an anchoring penalty to pull the student back toward the reference distribution. Under explicit regularity and step-size assumptions, a theoretical analysis establishes subsequential primal stationarity and a Polyak-Łojasiewicz-based tracking rate. Experiments across model families with different architectures and scales show that LAMP improves aggregate multi-domain performance over strong MOPD baselines under different settings.

1 INTRODUCTION

The rapid advancement of Large Language Models (LLMs) has necessitated efficient post-training paradigms to align and compress their vast capabilities (Aryan et al., 2023). Knowledge Distillation (KD) has emerged as a fundamental technique for this alignment (Hinton et al., 2015). However, conventional off-policy methods, which pull the student toward a fixed external target distribution using ground truth responses or teacher-generated outputs, exhibit uniform, aggressive token-level gradients that are prone to catastrophic forgetting and capability degradation (Kim & Rush, 2016; Ko et al., 2024). Because these methods optimize over a static sampling distribution rather than the student's own generated trajectories, they fail to properly address the train-inference distribution mismatch, thereby capping practical performance ceilings well below those of the teacher models (Song & Zheng, 2026).

To overcome the limitations of off-policy training, On-Policy Distillation (OPD) was introduced to optimize students based on self-generated rollouts scored by a teacher (Agarwal et al., 2024), dynamically matching the teacher's distribution via reverse Kullback-Leibler (KL) divergence (Gu et al., 2023). Building upon this, Multi-Teacher On-Policy Distillation (MOPD) scales this approach to integrate heterogeneous domain specialists into a unified student model, serving the critical purposes of capability merging to preserve peak expert performance across disparate tasks, and capability recovery to heal degradation induced by sequential reinforcement learning (Xiao et al., 2026; Yang et al., 2026b). Despite these advances, existing MOPD frameworks predominantly manage the optimization process heuristically through fixed parameter scaling or hard prompt routing (Yang et al., 2026a). Consequently, they lack a dynamic, mathematically rigorous constraint mechanism capable of automatically adjusting to parameter volatility, leaving the student vulnerable to capability regression when exposed to aggressive teacher pools.

The vulnerability of current MOPD frameworks becomes apparent when viewed through a geometric lens (wh., 2026; Brown & Opus, 2026). As illustrated in Fig. 1 (a), traditional MOPD approaches

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.

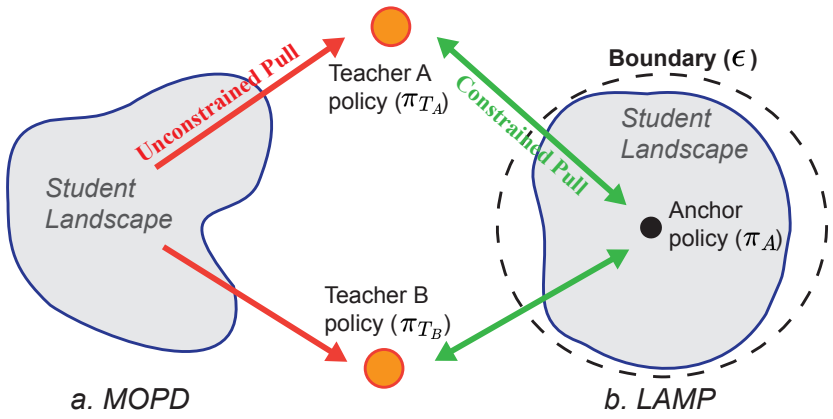


Figure 1: **A geometric comparison of policy updates in MOPD. (a) Traditional MOPD:** The student landscape is subjected to unconstrained gradient pulls toward disparate specialist teachers (π_{T_A} and π_{T_B}). These aggressive, biased updates can cause catastrophic distributional drift, inadvertently eroding previously established foundational capabilities. **(b) LAMP:** The dynamic dual mechanism increases the penalty on divergence from the static anchor policy (π_A) when the estimated KL exceeds the budget ϵ, encouraging specialist updates that preserve baseline behavior.

produce dense, biased per-token gradients that are highly concentrated toward individual specialist teachers. When these unconstrained gradient updates conflict across multiple domains, they inadvertently shift probability mass away from foundational behaviors, eroding previously established capabilities without a built-in mechanism to distinguish task-critical updates from incidental artifacts.

To bridge this critical gap, we propose Lagrangian Anchored Multi-Teacher Post-Training (**LAMP**), an adaptive framework that dynamically guards student generalization during distillation. Specifically, LAMP formulates a composite dual advantage that modulates parameter updates in real time, structurally balancing specialist capability acquisition with foundational capability preservation. In contrast to unconstrained dynamics, Fig. 1 (b) illustrates LAMP's anchored distributional geometry. By anchoring the policy update to an immutable static model, a running dual variable adjusts the penalty on divergence from the reference distribution. Our conditional analysis characterizes primal stationarity and tracking of the regularized objective under explicit assumptions; the experiments assess finite-run training behavior. Empirically, we validate LAMP through extensive experiments across multiple open-source model families and diverse evaluation domains. Our results demonstrate that LAMP effectively harmonizes specialized capability acquisition with the preservation of foundational knowledge. Loss-landscape analysis suggests flatter minima for LAMP, while evaluations across architectures, model sizes, and training mixtures show a stronger aggregate capability trade-off than competitive baselines.

2 Related Work

On-Policy Distillation. It has emerged as an effective post-training primitive for mitigating the train-inference distribution mismatch inherent in auto-regressive text generation (Song & Zheng, 2026). Generalized Knowledge Distillation (GKD) established the core paradigm of training students on self-generated rollouts scored by a teacher (Agarwal et al., 2024). To prevent the over-allocation of probability mass to sparse teacher modes, MiniLLM adopted mode-seeking reverse Kullback-Leibler (KL) divergence (Gu et al., 2023). Subsequent works improved optimization stability and efficiency; DistiLLM introduced skew divergences with an adaptive replay buffer (Ko et al., 2024), and DistiLLM-2 applied asymmetric contrastive losses tailored to distinct teacher and student trajectories (Ko et al., 2025). Despite these methodological advances, existing single-teacher OPD paradigms are intrinsically limited in multi-task scenarios. They assume a unified optimization trajectory and lack explicit regulatory boundaries to counter parameter-space interference or baseline capability drift when presented with conflicting feedback signals.

Multi-Teacher On-Policy Distillation. To address the intrinsic limitations of single-teacher frameworks and resolve the severe multi-domain "see-saw" effect, contemporary research has scaled these methods into Multi-Teacher On-Policy Distillation (MOPD) architectures. A primary application of MOPD is capability merging, where frameworks such as MiMo (Xiao et al., 2026) and DeepSeek-V4 (DeepSeek-AI, 2026) integrate heterogeneous domain specialists into a unified student, preserving peak expert performance across disparate tasks. Conversely, MOPD is also deployed for forgetting recovery; models like GLM-5 (Zeng et al., 2026) and Nemotron-Cascade 2 (Yang et al., 2026b) utilize cross-stage distillation from terminal or intermediate pipeline checkpoints to heal capability degradation induced by sequential reinforcement learning. Furthermore, recent theoretical generalizations, such as Generalized OPD, demonstrate that reward extrapolation (ExOPD) can successfully push a student model past its teachers' performance boundaries by scaling implicit reward signals (Yang et al., 2026b). However, while these approaches effectively navigate multi-domain alignment via fixed parameter scaling, outcome-reward augmentation, or hard prompt routing, they continue to manage the optimization process heuristically. They lack a dynamic, mathematically rigorous constraint mechanism capable of automatically adjusting to parameter volatility, leaving the student model vulnerable to capability regression when exposed to imbalanced datasets or aggressive teacher pools. We bridge this gap by formalizing an adaptive Lagrangian anchoring boundary whose penalty responds to observed divergence from the SFT anchor.

3 Problem Formulation

Let $\mathcal{D}$ denote a multi-domain prompt dataset where each input sequence $x \in \mathcal{D}$ maps to a specific domain label or routing token targeting a designated teacher within a specialized pool $\mathcal{P} = \{\pi_1, \pi_2, \ldots, \pi_K\}$. We denote the selected teacher for a given prompt x using the shortened notation $\pi_T \in \mathcal{P}$. In Multi-Teacher On-Policy Distillation (MOPD), trajectories $y = (y_1, y_2, \ldots, y_{|y|})$ are sampled directly from the active student policy $\pi_\theta(\cdot \mid x)$. The objective aligns the student's on-policy token distributions with the corresponding specialist teacher via a token-level reverse Kullback-Leibler (KL) divergence. Formally, the target distillation loss $\mathcal{L}_T(\theta)$ is defined as:

$$\mathcal{L}_T(\theta) = \mathbb{E}_{\substack{x\sim\mathcal{D} \\ y\sim\pi_\theta}}\Big[\frac{1}{|y|}\sum_{t=1}^{|y|} D_{\mathrm{KL}}\big(\pi_\theta(\cdot \mid x, y_{<t}) \,\|\, \pi_T(\cdot \mid x, y_{<t})\big)\Big]. \tag{1}$$

On sampled student trajectories, we use a token-level update direction that treats sampled prefixes as fixed and scales each score by a stopped advantage (Xiao et al., 2026):

$$g_T(\theta) = -\mathbb{E}_{x,y}\Big[\frac{1}{|y|}\sum_{t=1}^{|y|} \hat{A}_t^T \cdot \nabla_\theta \log \pi_\theta(y_t \mid x, y_{<t})\Big], \tag{2}$$

where the target advantage operator $\hat{A}_t^T$ is defined as the log-probability ratio between the teacher and student policies (with the stop-gradient deferred to the composite dual advantage) (Xiao et al., 2026):

$$\hat{A}_t^T = \log \pi_T(y_t \mid x, y_{<t}) - \log \pi_\theta(y_t \mid x, y_{<t}).$$

While this formulation provides dense, per-token directional supervision (Brown & Opus, 2026), computing gradients over mixed batches exposes the student policy to massive gradient conflict in parameter space. Steps driven by an aggressive specialist teacher can unintentionally erase foundational knowledge or impair performance in alternative domains, leading to severe capability regression.

4 Methodology

4.1 Motivation

To address the issues of severe multi-domain gradient conflicts in Multi-Teacher On-Policy Distillation (MOPD), previous approaches such as (Xiao et al., 2026) introduce a self-model into the teacher pool to regularize training. Inspired by reinforcement learning practices, such as Trust Region Policy Optimization (Schulman et al., 2015) and Proximal Policy Optimization (Schulman et al., 2017),

Algorithm 1 LAMP: Lagrangian Anchored Multi-Teacher Post-Training (Simplified)

Input: Multi-domain dataset $\mathcal{D}$; raw student π_{θ_0}; teacher pool $\mathcal{P}$; divergence budget ϵ; learning rates $\eta_\theta, \eta_\lambda$
Output: Distilled student π_{θ^*}

Freeze the anchor: $\pi_A \leftarrow \pi_{\theta_0}$; set $\theta \leftarrow \theta_0$, $\lambda \leftarrow 0.1$;
for each training step **do**
  Sample prompts $(x, \pi_T) \sim \mathcal{D}$ and roll out on-policy: $y \sim \pi_\theta(\cdot \mid x)$;
  Teacher advantage: $\hat{A}_t^T \leftarrow \log \pi_T(y_t \mid x, y_{<t}) - \log \pi_\theta(y_t \mid x, y_{<t})$;
  Anchor advantage: $\hat{A}_t^A \leftarrow \log \pi_\theta(y_t \mid x, y_{<t}) - \log \pi_A(y_t \mid x, y_{<t})$;
  Composite dual advantage: $\hat{A}_t^D \leftarrow \mathrm{sg}\left[\hat{A}_t^T\right] - \lambda \cdot \mathrm{sg}\left[\hat{A}_t^A\right]$;
  Primal update: $\theta \leftarrow \theta + \eta_\theta \cdot \frac{1}{|y|} \sum_{t=1}^{|y|} \hat{A}_t^D \cdot \nabla_\theta \log \pi_\theta(y_t \mid x, y_{<t})$;
  Dual update: $\lambda \leftarrow \max\left\{0,\ \lambda + \eta_\lambda \cdot (\mathcal{L}_A(\theta) - \epsilon)\right\}$;
end for

where a reference model is introduced to prevent gradient explosion and constrain policy updates, we address this vulnerability by introducing an immutable baseline anchor model π_A. Rather than introducing a static penalty that continuously diminishes tracking efficiency, we define a dynamic constraint boundary using an adaptive Lagrangian dual framework.

To operationalize this principle, we introduce Lagrangian Anchored Multi-Teacher Post-Training (**LAMP**), a streamlined algorithm that dynamically guards student generalization during distillation. At each training step, the algorithm computes two distinct, token-level distributional distances: the capability acquisition step relative to the routed specialist teacher π_T and the capability preservation step relative to the static anchor π_A. LAMP structurally integrates these signals into a composite dual advantage that modulates parameter updates in real time, automatically scaling up the anchoring constraint whenever a specialized gradient update threatens to distort the student's foundational representations. As outlined in Algorithm 1, LAMP continuously manages the post-training loop by executing online, on-policy rollouts and maintaining a running dual variable λ as an automated thermal regulator; the complete specification is provided in Algorithm 2 (Appendix A).

4.2 Adaptive Lagrangian Multi-Domain Anchoring

To limit policy drift under specialist updates, we introduce a divergence budget relative to the baseline anchor π_A, which we define as the raw student checkpoint prior to distillation. We formulate this constrained alignment problem as:

$$\min_\theta \mathcal{L}_T(\theta) \quad \textit{sub. to} \quad \mathcal{L}_A(\theta) \le \epsilon,$$

where $\mathcal{L}_A(\theta)$ represents the dense reverse KL divergence between the active student policy and the static anchor reference model:

$$\mathcal{L}_A(\theta) = \mathbb{E}_{\substack{x \sim \mathcal{D} \\ y \sim \pi_\theta}} \Big[\frac{1}{|y|} \sum_{t=1}^{|y|} D_{\mathrm{KL}}\Big(\pi_\theta(\cdot \mid x, y_{<t}) \,\|\, \pi_A(\cdot \mid x, y_{<t})\Big) \Big]. \tag{3}$$

Introducing the Lagrange multiplier $\lambda \ge 0$ yields the Lagrangian:

$$\mathcal{L}_D(\theta, \lambda) = \mathcal{L}_T(\theta) + \lambda \cdot (\mathcal{L}_A(\theta) - \epsilon)\,.$$

We optimize $\mathcal{L}_D$ via alternating primal-dual updates. Combining the target and anchor terms gives the token-level update direction:

$$g_D(\theta, \lambda) = -\mathbb{E}_{x,y} \Big[\frac{1}{|y|} \sum_{t=1}^{|y|} \hat{A}_t^D \cdot \nabla_\theta \log \pi_\theta(y_t \mid x, y_{<t}) \Big], \tag{4}$$

where $s_t = (x, y_{<t})$ denotes the context state prefix at token step t. The composite dual advantage $\hat{A}_t^D$ is defined as:

$$\hat{A}_t^D = \mathrm{sg}[\hat{A}_t^T] - \lambda \cdot \mathrm{sg}[\hat{A}_t^A],$$

where $\hat{A}_t^A = \log \pi_\theta(y_t \mid s_t) - \log \pi_A(y_t \mid s_t)$ denotes the anchor advantage, representing the token-level distributional shift relative to the static anchor model. The stopped-advantage direction holds sampled prefixes fixed; it need not equal the full gradient of the on-policy trajectory objective, which also differentiates the prefix distribution.

Concurrently, the dual variable λ is updated by projected gradient ascent in response to constraint violation:

$$\lambda \leftarrow \max\big\{0, \lambda + \eta_\lambda \cdot (\mathcal{L}_A(\theta) - \epsilon)\big\},$$

where η_λ is the dual learning rate. This mechanism yields a self-correcting optimization loop. If a specialist teacher forces an update that significantly deviates from the anchor distribution, $\mathcal{L}_A(\theta)$ spikes above the threshold ϵ, increasing λ and immediately amplifying the anchor penalty in the token advantage to counteract further drift.

5 Theoretical Analysis

LAMP couples specialist alignment to reference-policy KL through a dual weight that changes with anchor divergence. We analyze how this moving penalty interacts with primal descent in an idealized stochastic-gradient model of the population Lagrangian, drawing on constrained optimization analysis (Shing et al., 2025). The first result concerns non-convex objectives; the second studies how closely the primal iterate follows the regularized objective as the dual weight changes.

Theorem 5.1 ***(Subsequential Primal Stationarity under Non-convexity)*** *Assume $\epsilon \geq 0$, $\mathcal{L}_D(\cdot, \lambda)$ is uniformly L-smooth for the dual iterates, the primal stochastic gradient is conditionally unbiased with bounded second moment G^2, and $|\mathcal{L}_A(\theta) - \epsilon| \leq B$ on a fixed region containing the iterates, their trial updates, and the connecting segments. Let the idealized dual update use the population residual $\mathcal{L}_A(\theta_k) - \epsilon$. If $\sum_{k=1}^{\infty} \eta_{\theta,k} = \infty$, $\sum_{k=1}^{\infty} \eta_{\theta,k}^2 < \infty$, and $\sum_{k=1}^{\infty} \eta_{\lambda,k} < \infty$, then primal descent with projected dual ascent satisfies* $\liminf_{k\to\infty} \mathbb{E}[\|\nabla_\theta \mathcal{L}_D(\theta_k, \lambda_k)\|^2] = 0$.

Theorem 5.1 shows that the expected squared primal gradient becomes arbitrarily small along a subsequence. Under its assumptions, changing the anchor weight is therefore compatible with subsequential stationarity of the teacher-and-anchor objective.

To quantify primal tracking as λ changes, we impose a uniform Polyak-Łojasiewicz (PL) condition on $\mathcal{L}_D(\cdot, \lambda)$. It bounds objective suboptimality by the squared gradient norm without requiring convexity.

Theorem 5.2 ***(Primal Tracking under the PL Condition)*** *Under the assumptions of Theorem 5.1, suppose $\mathcal{L}_D(\cdot, \lambda)$ satisfies the uniform μ-PL condition $\frac{1}{2}\|\nabla_\theta \mathcal{L}_D(\theta, \lambda)\|^2 \geq \mu(\mathcal{L}_D(\theta, \lambda) - \mathcal{L}_D(\theta_\lambda^*, \lambda))$, where θ_λ^* minimizes $\mathcal{L}_D(\cdot, \lambda)$ in the same fixed region. For $\eta_{\theta,k} = c/(k + K_0)$ with $c > 1/(2\mu)$ and sufficiently large K_0, and $\eta_{\lambda,k} = O(\eta_{\theta,k}^2)$, the idealized updates satisfy* $\mathbb{E}[\mathcal{L}_D(\theta_k, \lambda_k) - \mathcal{L}_D(\theta_{\lambda_k}^*, \lambda_k)] = O(1/k)$.

Theorem 5.2 gives an $O(1/k)$ expected gap to the minimum of the regularized objective at the current λ_k. For LAMP, this characterizes how primal optimization can follow a changing balance between specialist alignment and the static anchor when the dual update is sufficiently slow. Together, the two results describe the optimization role of adaptive anchoring; proofs and their scope are given in Appendix B.

6 Experiments

6.1 Experimental Setup

Base Models. We evaluate LAMP across three model families spanning distinct architectures and pre-training paradigms: Qwen2 (7B→1.5B) (Yang et al., 2024), Qwen2.5 (7B→1.5B) (Yang et al., 2025b), and Gemma 2 (9B→2B) (Team et al., 2024). These families cover diverse scales (1.5B–9B), ensuring robust evaluation across varied distillation scenarios.

Table 1: Multi-domain capability merging results under balanced data conditions.

			Instruction Following (WR %)			Math Reasoning (Pass@1)			Code Generation (Pass@1)		
M_t	M_s	Method	*Evol-Inst*	*UltraFeed*	*Avg.*	*MATH*	*GSM8K*	*Avg.*	*HumanEval*	*MBPP*	*Avg.*
Qwen2 (7B)	Qwen2 (1.5B)	M_T	67.3/69.5	64.1/67.3	65.7/68.4	41.1/46.8	81.3/83.6	61.2/65.2	81.1/84.6	67.7/68.4	74.4/76.5
		M_S	17.9	21.5	19.7	21.6	46.9	34.3	34.8	46.9	40.8
		SFT	26.4	35.3	30.8	23.5	50.7	37.1	37.4	47.2	42.3
		OPD	42.2	55.3	48.8	25.1	55.2	40.2	41.0	49.5	45.3
		MOPD	48.3	58.3	53.3	27.5	58.9	43.2	43.1	50.8	47.0
		ExOPD	44.1	56.2	50.2	**30.4**	**61.3**	**45.8**	**45.2**	50.4	47.8
		LAMP	**50.8**	**59.6**	**55.2**	29.9	60.8	45.4	44.8	**51.3**	**48.1**
Qwen2.5 (7B)	Qwen2.5 (1.5B)	M_T	71.0/73.7	72.3/75.4	71.7/74.6	52.5/54.6	86.1/87.3	69.3/71.0	58.3/60.0	75.2/77.8	66.8/68.9
		M_S	23.7	25.4	24.6	35.0	68.5	51.8	37.2	60.2	48.7
		SFT	27.7	37.1	32.4	25.1	55.0	40.1	42.1	63.7	52.9
		OPD	44.1	57.9	51.0	27.5	59.7	43.6	44.5	64.1	54.3
		MOPD	51.2	60.6	55.9	30.1	63.2	46.7	47.9	65.3	56.6
		ExOPD	46.8	59.5	53.2	32.3	**65.8**	**49.0**	**51.7**	65.9	**58.8**
		LAMP	**52.7**	**63.2**	**57.9**	**32.6**	64.9	48.8	50.8	**66.2**	58.5
Gemma2 (9B)	Gemma2 (2B)	M_T	84.3/85.1	82.4/83.9	83.4/84.5	39.4/41.3	72.3/74.1	55.9/57.7	38.7/40.5	63.6/65.9	51.2/53.2
		M_S	16.8	26.5	21.7	18.3	30.3	24.3	19.5	42.1	30.8
		SFT	31.9	53.7	42.8	21.5	34.6	28.1	25.8	43.5	34.7
		OPD	48.3	74.3	61.3	23.8	41.8	32.8	28.9	44.7	36.8
		MOPD	49.9	75.3	62.6	27.4	47.4	37.4	35.6	46.3	41.0
		ExOPD	45.8	71.4	58.6	**29.3**	51.2	40.3	38.7	**48.4**	43.6
		LAMP	**51.8**	**77.3**	**64.6**	28.5	**53.8**	**41.2**	**40.1**	47.5	**43.8**

Note: M_T and M_S denote teacher and base student performance, respectively. For each M_T, the former number indicates the performance of the single generalist teacher used for OPD, while the latter number represents the performance of the domain-specialist teacher pool (three specialized models for IF, Math, and Code) used for MOPD, ExOPD, and LAMP. All distillation methods are initialized from the Generalist SFT Checkpoint and trained for 3 epochs. Instruction Following is evaluated via Winning Rate (%) with LLM-as-a-Judge against `gpt-3.5-turbo`; Math Reasoning and Code Generation are evaluated via Pass@1 with greedy decoding. Best student results within each model family are **bold**.

Training. We construct a 60k multi-domain mixture comprising 20k Instruction Following (IF) prompts from UltraChat200k (Ding et al., 2023), 20k Math reasoning prompts from MetaMathQA (Yu et al., 2024), and 20k Code generation prompts from WizardCoder (Luo et al., 2024). Training proceeds in two phases. **Phase 1**: Supervised Fine-Tuning (SFT) on the full mixture produces the *Generalist SFT Checkpoint*, serving as the shared initialization. **Phase 2**: On-Policy Distillation variants are applied, including OPD, MOPD, ExOPD (MOPD with reward extrapolation $\alpha = 1.25$ and $\pi_{\text{ref}} =$ SFT Checkpoint), and LAMP (MOPD with Lagrangian anchoring, $\pi_A =$ SFT Checkpoint, $\lambda_0 = 0.1$, $\epsilon = 0.15$). Full hyperparameters are detailed in Appendix C.

Evaluation. We evaluate across three domains: (1) **IF**: Evol-Instruct (Xu et al., 2023) and UltraFeedback (Cui et al., 2023) via Winning Rate (WR) using LLM-as-a-Judge (GPT-4o/GPT-4o-mini) against `gpt-3.5-turbo` (Zheng et al., 2023). (2) **Math**: GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021) via Pass@1 metric. (3) **Code**: HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021) via Pass@1 using the EvalPlus (Chen et al., 2021) framework.

Baselines. The main comparison uses four baselines: (1) SFT (Phase 1 checkpoint); (2) OPD (Gu et al., 2023) (single-teacher); (3) MOPD (DeepSeek-AI, 2026) (three domain specialists); and (4) ExOPD (Yang et al., 2026b) (MOPD with reward extrapolation). The ablation additionally includes FixedMix-MOPD, which uses a fixed mixture of each specialist and the SFT anchor.

6.2 Balanced Multi-Domain Capability Merging

We evaluate the methods on a balanced mixture of 20k IF, 20k Math, and 20k Code prompts to examine whether one student can acquire gains across all three domains. Tab. 1 reports the benchmark scores and domain averages for each model family.

Multi-teacher guidance improves all three domains. Relative to single-teacher OPD, MOPD raises every domain average in the three families, with an increase of about 3 points in the unweighted three-domain mean in each case. On Gemma 2, for example, Math and Code rise from 32.8 and 36.8 under OPD to 37.4 and 41.0 under MOPD, while IF also increases from 61.3 to 62.6. The

comparison shows the value of domain-specialist supervision under the same prompt mixture and student initialization.

LAMP extends the gains over MOPD. Its IF average increases by 1.9–2.0 points across families, while Math and Code improve by 2.1–3.8 and 1.1–2.8 points, respectively. The IF gains accompany improvements in both specialist domains. On Gemma 2, the largest Math and Code gains in this comparison raise the respective averages from 37.4 to 41.2 and from 41.0 to 43.8.

The comparison with ExOPD is more selective. Averaging the three domain scores, LAMP leads ExOPD by 1.6, 1.4, and 2.4 points on Qwen2, Qwen2.5, and Gemma 2. It improves both IF benchmarks in every family, giving IF-average gains of 5.0, 4.7, and 6.0 points. The specialist results are closer: ExOPD leads Qwen2 Math by 0.4 points and Qwen2.5 Math and Code by 0.2 and 0.3 points. On Qwen2.5, LAMP gains 0.3 points on MATH and MBPP but trails by 0.9 on GSM8K and HumanEval. Gemma 2 shows the same unevenness: LAMP gains 6.0 and 5.9 points on Evol-Instruct and UltraFeedback and 2.6 on GSM8K, yet trails by 0.8 on MATH and 0.9 on MBPP. The aggregate lead therefore reflects a repeatable IF advantage alongside smaller, benchmark-dependent specialist differences.

6.3 Robustness to Imbalanced Mixtures with Larger Models

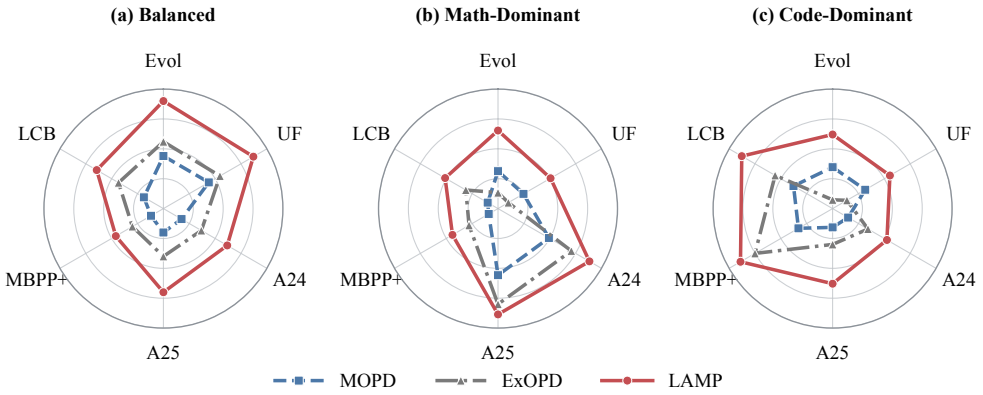


Figure 2: **Performance under balanced and imbalanced mixtures.** MOPD, ExOPD, and LAMP distill Qwen3-14B into Qwen3-8B (both non-thinking) using 60k IF/Math/Code prompts: **(a)** balanced (20k/20k/20k), **(b)** Math-Dominant (10k/40k/10k), and **(c)** Code-Dominant (10k/10k/40k). IF is winning rate; Math and Code are Pass@1 (%). Clockwise spokes are Evol-Instruct (Evol, 37–52), UltraFeedback (UF, 43–56), AIME'24 (A24, 35–47), AIME'25 (A25, 27–34), MBPP+ (66–76), and LiveCodeBench v5 (LCB, 33–42). Each benchmark uses the same center-to-rim range in all panels.

To assess LAMP on a larger student and under imbalanced domain mixtures, we distill Qwen3-14B (non-thinking) into Qwen3-8B (non-thinking) (Yang et al., 2025a) using the balanced, Math-Dominant, and Code-Dominant 60k mixtures in Fig. 2. We evaluate IF on Evol-Instruct and UltraFeedback (Xu et al., 2023; Cui et al., 2023), Math on AIME'24 and AIME'25 (Zhang & Math-AI Team, 2024; 2025), and Code on MBPP+ and LiveCodeBench v5 (Liu et al., 2023; Jain et al., 2024). All three methods use the same mixture within each panel.

LAMP exceeds ExOPD on all six reported benchmarks in each mixture, but the margins vary. Its unweighted six-benchmark mean exceeds ExOPD's by 3.0 points in the balanced mixture, 3.2 in Math-Dominant, and 3.7 in Code-Dominant. The largest differences under skew are in IF: averaging the two IF benchmarks, LAMP leads ExOPD by 6.6 points in Math-Dominant and 6.8 in Code-Dominant, compared with 4.7 under balanced training. Both methods lose IF performance as the mixture shifts toward a specialist, yet LAMP's drop from its balanced IF average is 4.2 points in each skewed setting, versus 6.1–6.3 for ExOPD.

The specialist-domain differences are more modest. In Math-Dominant training, LAMP gains 2.1 points on AIME'24 but only 0.6 on AIME'25 over ExOPD. In Code-Dominant training, its gains are 1.4 on MBPP+ and 2.9 on LiveCodeBench v5. Within the tested mixtures, LAMP's advantage is

greatest when a Math- or Code-heavy prompt stream makes IF retention the priority. The results are consistent with adaptive anchoring limiting drift, although benchmark scores alone do not isolate that mechanism.

7 Analysis and Discussion

7.1 Ablation Study

We use the Qwen2-1.5B ablation to assess the roles of specialist supervision, the SFT anchor, and adaptive control of the anchor penalty in LAMP. Tab. 2 compares these components under the balanced 60k mixture. All variants share the SFT initialization, training data, epochs, and other hyperparameters.

Table 2: **Anchor-control ablation on Qwen2-1.5B.** IF is average winning rate; Math and Code are average Pass@1 (%). Overall is the mean of the three domain scores. MOPD corresponds to $\lambda = 0$; FixedMix uses $\rho = 0.3$.

	OPD	MOPD	ExOPD	Fixed λ			FixedMix	LAMP
		0		0.1	0.5	1.0	$\rho = 0.3$	Dynamic
IF	48.8	53.3	50.2	51.4	54.8	50.5	54.4	**55.2**
Math	40.2	43.2	**45.8**	45.1	44.6	43.5	44.8	45.4
Code	45.3	47.0	47.8	44.9	46.4	47.5	47.5	**48.1**
Overall	44.8	47.8	47.9	47.1	48.6	47.2	48.9	**49.6**

Specialist supervision. Removing the anchor penalty gives MOPD ($\lambda = 0$), which raises the three-domain mean from 44.8 under single-teacher OPD to 47.8. IF, Math, and Code increase by 4.5, 3.0, and 1.7 points, respectively. This establishes the contribution of routed specialists before anchor control is introduced.

Static anchoring. A fixed penalty changes the domain trade-off, but its effect depends on the chosen weight. At $\lambda = 0.1$, Math rises to 45.1 while IF and Code fall below MOPD to 51.4 and 44.9. Increasing λ to 1.0 raises Code to 47.5 but lowers Math to 43.5, and IF peaks at the intermediate value of 0.5. That value gives the best fixed-penalty mean, 48.6. FixedMix-MOPD instead replaces each specialist target with $\tilde{\pi}_T(\cdot \mid s) = (1-\rho)\pi_T(\cdot \mid s) + \rho\pi_A(\cdot \mid s)$ at $\rho = 0.3$. Its mean of 48.9, with gains over MOPD in all three domains, shows that including the SFT anchor in a fixed target mixture also helps on this setting.

Adaptive penalty. LAMP uses the anchor divergence to update λ rather than holding the penalty fixed. It reaches a mean of 49.6, exceeding FixedMix-MOPD by 0.7 points and the best fixed-penalty setting by 1.0 point. Against the best fixed λ selected separately for each domain, LAMP gains 0.4 points on IF, 0.3 on Math, and 0.6 on Code. These small but simultaneous gains support the value of adapting the penalty beyond adding an anchor alone.

7.2 Optimization Dynamics and Training Stability

Fig. 3 (a, b) contrasts the training trajectories of unconstrained MOPD and LAMP on GSM8K (Math) and MBPP (Code). The dashed lines show the performance of the respective teacher models. MOPD exhibits non-monotonic behavior, including a stagnation period between steps 400 and 600 on GSM8K and performance dips at steps 200 and 600 on MBPP. LAMP shows smoother trajectories in these runs and narrows the gap to the teacher models. At step 600, LAMP leads MOPD by 5.2 points on GSM8K and 2.1 on MBPP, whereas the final margins are 1.9 and 0.5 points. MOPD briefly exceeds LAMP on both curves at step 800. The contrast is therefore strongest during training, with a smaller endpoint gain on MBPP. Both methods use the same initialization and data mixture, while LAMP adjusts the anchor penalty in $\hat{A}_t^D$ from the measured KL residual. A trace of λ alongside $\mathcal{L}_A(\theta) - \epsilon$ would show more directly when the dual feedback responds, and repeated training runs would quantify how variable the trajectories are.

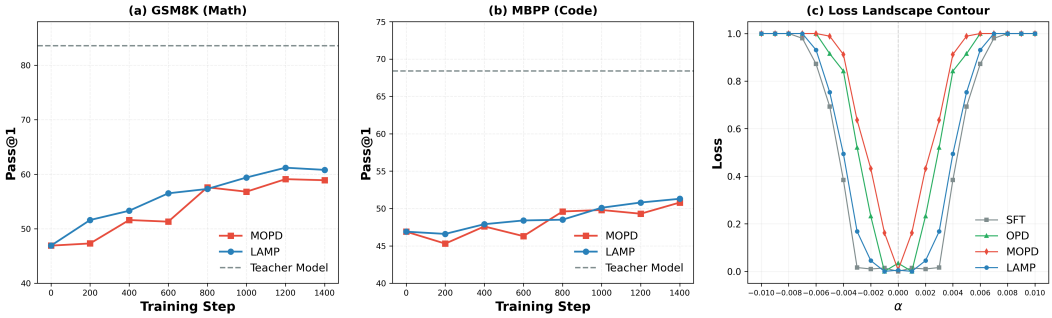


Figure 3: **Optimization dynamics and loss landscape geometries. (a, b)** Training trajectories on GSM8K (Math) and MBPP (Code). Dashed lines indicate teacher performance. LAMP shows smoother trajectories in these runs, while MOPD exhibits performance dips (e.g., at steps 200 and 600 in MBPP). **(c)** One-dimensional loss-landscape projections for Qwen2-1.5B. LAMP has a wider basin along the sampled directions than MOPD and OPD, approaching the pre-distillation SFT profile.

7.3 Loss-Landscape and Parameter Sensitivity

Fig. 3 (c) visualizes one-dimensional projections of the discrete 0-1 loss landscape against perturbation magnitude α along random normalized parameter directions (Chen et al., 2025). The pre-distillation SFT checkpoint has the widest projected basin. Among the distillation methods, MOPD has the narrowest basin, OPD is slightly wider, and LAMP approaches the SFT profile. The comparison is informative alongside Tab. 1: on Qwen2, SFT scores 30.8, 37.1, and 42.3 on IF, Math, and Code, whereas LAMP reaches 55.2, 45.4, and 48.1. LAMP thus combines gains beyond the SFT checkpoint with lower sensitivity than MOPD along the sampled directions. The wider SFT basin shows why this local measure must be read alongside task scores.

8 Conclusion

In this work, we introduced LAMP, a Lagrangian approach to addressing multi-domain gradient conflicts in Multi-Teacher On-Policy Distillation. By transitioning from heuristic optimization constraints to an adaptive dual framework, LAMP balances specialized capability acquisition with the preservation of foundational representations. Our framework utilizes a running dual variable λ that dynamically scales anchoring penalties based on measured divergence relative to an immutable reference policy π_A.

Under the stated assumptions, the analysis establishes subsequential primal stationarity and tracking of the current regularized objective. The experiments show smoother training trajectories and improved aggregate performance in the settings studied. The larger-model imbalance experiment finds the clearest advantage in instruction-following retention when training favors a specialist domain, while gains on specialist benchmarks remain modest. LAMP thus offers an adaptive way to balance capability merging against drift from the reference policy.

AI use statement

In this work, generative AI was used to polish the manuscript, correct grammatical errors, and assist with coding and debugging for experiments via Codex. The authors take full responsibility for the final manuscript, code, results, and claims.

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*, 2024.

Abi Aryan, Aakash Kumar Nain, Andrew McMahon, Lucas Augusto Meyer, and Harpreet Singh Sahota. The costly dilemma: Generalization, evaluation and cost-optimal deployment of large language models. *arXiv preprint arXiv:2308.08061*, 2023.

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

Will Brown and Claude Opus. On sft, rl, and on-policy distillation: A technical guide. `https://claude.ai/public/artifacts/478fd2c4-e37d-4f1e-8295-780d811b1859`, 2026. Accessed: 2026-05-23.

Huanran Chen, Yinpeng Dong, Zeming Wei, Yao Huang, Yichi Zhang, Hang Su, and Jun Zhu. Unveiling the basin-like loss landscape in large language models. *arXiv preprint arXiv:2505.17646*, 2025.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

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

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, et al. Ultrafeedback: Boosting language models with scaled ai feedback. *arXiv preprint arXiv:2310.01377*, 2023.

DeepSeek-AI. Deepseek-v4: Towards highly efficient million-token context intelligence, 2026.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. Enhancing chat language models by scaling high-quality instructional conversations. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 3029–3051, 2023.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models. *arXiv preprint arXiv:2306.08543*, 2023.

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. *arXiv preprint arXiv:2103.03874*, 2021.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

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

Yoon Kim and Alexander M Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1317–1327, 2016.

Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. Distillm: Towards streamlined distillation for large language models. *arXiv preprint arXiv:2402.03898*, 2024.

Jongwoo Ko, Tianyi Chen, Sungnyun Kim, Tianyu Ding, Luming Liang, Ilya Zharkov, and Se-Young Yun. Distillm-2: A contrastive approach boosts the distillation of llms. *arXiv preprint arXiv:2503.07067*, 2025.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? rigorous evaluation of large language models for code generation, 2023. URL https://arxiv.org/abs/2305.01210.

Ziyang Luo, Can Xu, Pu Zhao, Qingfeng Sun, Xiubo Geng, Wenxiang Hu, Chongyang Tao, Jing Ma, Qingwei Lin, and Daxin Jiang. Wizardcoder: Empowering code large language models with evol-instruct. In *International Conference on Learning Representations*, 2024.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pp. 1889–1897. PMLR, 2015.

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

Makoto Shing, Kou Misaki, Han Bao, Sho Yokoi, and Takuya Akiba. Taid: Temporally adaptive interpolated distillation for efficient knowledge transfer in language models. *arXiv preprint arXiv:2501.16937*, 2025.

Mingyang Song and Mao Zheng. A survey of on-policy distillation for large language models. *arXiv preprint arXiv:2604.00626*, 2026.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

wh. Sft, rl, and on-policy distillation through a distributional lens. https://nrehiew.github.io/blog/sft_rl_opd/, 2026. Accessed: 2026-05-24.

Bangjun Xiao, Bingquan Xia, Bo Yang, Bofei Gao, Bowen Shen, Chen Zhang, Chenhong He, Chiheng Lou, Fuli Luo, Gang Wang, et al. Mimo-v2-flash technical report. *arXiv preprint arXiv:2601.02780*, 2026.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. Wizardlm: Empowering large pre-trained language models to follow complex instructions. *arXiv preprint arXiv:2304.12277*, 2023.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. Qwen2 technical report, 2024. URL https://arxiv.org/abs/2407.10671.

An Yang, Anfeng Li, Baosong Yang, et al. Qwen3 technical report, 2025a. URL https://arxiv.org/abs/2505.09388.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025b. URL https://arxiv.org/abs/2412.15115.

Wenkai Yang, Weijie Liu, Ruobing Xie, Kai Yang, Saiyong Yang, and Yankai Lin. Learning beyond teacher: Generalized on-policy distillation with reward extrapolation. *arXiv preprint arXiv:2602.12125*, 2026a.

Zhuolin Yang, Zihan Liu, Yang Chen, Wenliang Dai, Boxin Wang, Sheng-Chieh Lin, Chankyu Lee, Yangyi Chen, Dongfu Jiang, Jiafan He, et al. Nemotron-cascade 2: Post-training llms with cascade rl and multi-domain on-policy distillation. *arXiv preprint arXiv:2603.19220*, 2026b.

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 *International Conference on Learning Representations*, 2024.

Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, et al. Glm-5: From vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*, 2026.

Yifan Zhang and Math-AI Team. American invitational mathematics examination (AIME) 2024, 2024. URL https://huggingface.co/datasets/math-ai/aime24.

Yifan Zhang and Math-AI Team. American invitational mathematics examination (AIME) 2025, 2025. URL https://huggingface.co/datasets/math-ai/aime25.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623, 2023.

Algorithm 2 LAMP: Lagrangian Anchored Multi-Teacher Post-Training

Input: $\mathcal{D}$: Training dataset with prompt mappings (x, π_T); π_{θ_0}: Raw student model; $\mathcal{P}$: Teacher pool; ϵ: Divergence threshold; $\eta_\theta, \eta_\lambda$: Learning rates; B: Batch size; I: Max iterations
Output: π_{θ^*}: Distilled student model.

Initialize active policy parameters: $\theta \leftarrow \theta_0$;
Instantiate static anchor reference: $\pi_A \leftarrow \pi_{\theta_0}$;
Initialize Lagrange multiplier: $\lambda \leftarrow 0.1$;
for step $t = 1$ **to** I **do**
  Sample mini-batch $\mathcal{B} = \{(x_i, \pi_{T_i})\}_{i=1}^{B}$ from $\mathcal{D}$;
  Generate traces: $y_i \sim \pi_\theta(\cdot \mid x_i)$ for all $x_i \in \mathcal{B}$;
  Compute anchor log-probabilities: $\pi_A(y_{i,j} \mid x_i, y_{i,<j})$;
  Compute teacher log-probabilities: $\pi_{T_i}(y_{i,j} \mid x_i, y_{i,<j})$ for all $(x_i, \pi_{T_i}) \in \mathcal{B}$;
  Compute empirical anchor divergence:
  $\mathcal{L}_A(\theta) \leftarrow \frac{1}{B}\sum_{i=1}^{B} \frac{1}{|y_i|}\sum_{j=1}^{|y_i|} \log \frac{\pi_\theta(y_{i,j}|x_i, y_{i,<j})}{\pi_A(y_{i,j}|x_i, y_{i,<j})}$;
  for each sequence $i \in \mathcal{B}$ and position $j = 1$ **to** $|y_i|$ **do**
    $\hat{A}^T_{i,j} \leftarrow \log \pi_{T_i}(y_{i,j} \mid x_i, y_{i,<j}) - \log \pi_\theta(y_{i,j} \mid x_i, y_{i,<j})$; {Raw log-ratio; $\mathrm{sg}[\cdot]$ applied in composite advantage}
    $\hat{A}^A_{i,j} \leftarrow \log \pi_\theta(y_{i,j} \mid x_i, y_{i,<j}) - \log \pi_A(y_{i,j} \mid x_i, y_{i,<j})$;
    $\hat{A}^D_{i,j} \leftarrow \mathrm{sg}\left[\hat{A}^T_{i,j}\right] - \lambda \cdot \mathrm{sg}\left[\hat{A}^A_{i,j}\right]$;
  end for
  Compute policy gradient:
  $g_\theta \leftarrow -\frac{1}{B}\sum_{i=1}^{B} \frac{1}{|y_i|}\sum_{j=1}^{|y_i|} \hat{A}^D_{i,j} \cdot \nabla_\theta \log \pi_\theta(y_{i,j} \mid x_i, y_{i,<j})$;
  Update primal parameters: $\theta \leftarrow \theta - \eta_\theta \cdot g_\theta$;
  Update dual variable: $\lambda \leftarrow \max\left(0, \lambda + \eta_\lambda \cdot (\mathcal{L}_A(\theta) - \epsilon)\right)$;
end for
$\pi_{\theta^*} \leftarrow \pi_\theta$;

A Algorithmic Framework

This section presents the complete algorithmic specification of the LAMP framework introduced in Section 4, expanding the simplified overview of Algorithm 1. Algorithm 2 organizes the optimization into three sequential phases: (i) initialization of the static anchor model π_A from the raw student checkpoint and the dual variable $\lambda = 0.1$; (ii) a main training loop that performs on-policy rollouts, computes token-level log-probabilities under both the specialist teacher π_T and the anchor model, and assembles the composite dual advantage $\hat{A}^D_t = \mathrm{sg}[\hat{A}^T_t] - \lambda \cdot \mathrm{sg}[\hat{A}^A_t]$ to modulate gradient updates; and (iii) projected dual ascent, where λ responds to the measured divergence relative to the budget ϵ. This procedure operationalizes the Lagrangian anchoring framework described in Section 4.2, providing a self-contained specification for reproducible implementation.

Remark: At each iteration, the dual variable is updated using the pre-update estimate of $\mathcal{L}_A(\theta)$. The diminishing step-size schedules in the analysis below are separate from the finite-run schedule used in the experiments.

B Theoretical Analysis

These proofs assume unbiased gradients of the population Lagrangian, an exact dual residual, and the diminishing step sizes stated in the theorems. The stopped-prefix update and sampled residual in Algorithm 2 need not satisfy these oracle assumptions; the experiments also use cosine primal decay and a fixed dual rate. The results characterize primal stationarity and tracking under the stated conditions.

B.1 Definitions and Notation

To ensure clarity, we define the following symbols used throughout the analysis:

- θ : The set of active student policy parameters.
- π_A: The static anchor reference model.

- $\mathcal{L}_T(\theta)$: The target distillation loss aligning the student's on-policy token distributions with the specialist teacher π_T.
- $\mathcal{L}_A(\theta)$: The dense reverse KL divergence representing the capability preservation step relative to the static anchor.
- $\lambda \geq 0$: The Lagrange multiplier that continuously regulates the constraint boundary.
- $\mathcal{L}_D(\theta, \lambda) = \mathcal{L}_T(\theta) + \lambda \cdot (\mathcal{L}_A(\theta) - \epsilon)$: The Lagrangian objective.

B.2 Proof of Theorem 5.1

The objective is to establish the expected primal-gradient liminf under the assumptions of Theorem 5.1.

Step 1: Smoothness Assumption. Let $c(\theta) = \mathcal{L}_A(\theta) - \epsilon$. We assume that the population Lagrangian is uniformly L-smooth in θ over the relevant parameter region and dual iterates, so that:

$$\|\nabla_\theta \mathcal{L}_D(\theta_1, \lambda) - \nabla_\theta \mathcal{L}_D(\theta_2, \lambda)\| \leq L\|\theta_1 - \theta_2\|,$$

This is an explicit assumption; smooth activations and softmax alone do not imply uniformly bounded log-probabilities or gradients.

Step 2: Moment and Lower-Bound Conditions. Let $\mathcal{F}_k$ denote the history before the primal update. For the idealized estimator g_k, we assume $\mathbb{E}[g_k \mid \mathcal{F}_k] = \nabla_\theta \mathcal{L}_D(\theta_k, \lambda_k)$ and $\mathbb{E}[\|g_k\|^2 \mid \mathcal{F}_k] \leq G^2$. These properties are assumptions, rather than consequences of the stopped token advantage. Since $|c(\theta)| \leq B$, nonexpansiveness of the projection onto $[0, \infty)$ gives $|\lambda_{k+1} - \lambda_k| \leq B\eta_{\lambda,k}$. Summability of $\eta_{\lambda,k}$ therefore yields $0 \leq \lambda_k \leq \Lambda := \lambda_1 + B\sum_j \eta_{\lambda,j} < \infty$. For $\epsilon \geq 0$, nonnegativity of the KL losses then gives the uniform lower bound $\mathcal{L}_D(\theta_k, \lambda_k) \geq -\Lambda\epsilon$.

Step 3: Expected Primal Stationarity. Using L-smoothness for the primal update $\theta_{k+1} = \theta_k - \eta_{\theta,k} g_k$ gives:

$$\mathbb{E}[\mathcal{L}_D(\theta_{k+1}, \lambda_k) \mid \mathcal{F}_k] \leq \mathcal{L}_D(\theta_k, \lambda_k) - \eta_{\theta,k}\|\nabla_\theta \mathcal{L}_D(\theta_k, \lambda_k)\|^2 + \frac{LG^2\eta_{\theta,k}^2}{2}.$$

Because $\mathcal{L}_D$ is linear in λ and $|c(\theta)| \leq B$, the shift in the objective after the dual update is at most $B|\lambda_{k+1} - \lambda_k| \leq B^2\eta_{\lambda,k}$. This term must be retained when summing the descent inequalities.

Summing from $k = 1$ to N and taking expectations yields:

$$\sum_{k=1}^{N} \eta_{\theta,k}\mathbb{E}[\|\nabla_\theta \mathcal{L}_D(\theta_k, \lambda_k)\|^2] \leq \mathcal{L}_D(\theta_1, \lambda_1) + \Lambda\epsilon + \frac{LG^2}{2}\sum_{k=1}^{N}\eta_{\theta,k}^2 + B^2\sum_{k=1}^{N}\eta_{\lambda,k}.$$

The right-hand side is finite as $N \to \infty$, so:

$$\sum_{k=1}^{\infty} \eta_{\theta,k}\mathbb{E}[\|\nabla_\theta \mathcal{L}_D(\theta_k, \lambda_k)\|^2] < \infty.$$

Since $\sum_k \eta_{\theta,k} = \infty$, this implies:

$$\liminf_{k\to\infty} \mathbb{E}[\|\nabla_\theta \mathcal{L}_D(\theta_k, \lambda_k)\|^2] = 0.$$

The summable dual increments also imply that λ_k has a finite limit, completing the proof of Theorem 5.1.

B.3 Proof of Theorem 5.2

The objective is to bound the primal gap to the minimizer of $\mathcal{L}_D(\cdot, \lambda_k)$ under a uniform PL condition, without assuming strict convexity.

Step 1: Primal Descent Lemma for the Regularized Policy. Consider the stochastic gradient descent update rule for the primal objective at step k:

$$\theta_{k+1} = \theta_k - \eta_{\theta,k} g_{\theta_k},$$

where g_{θ_k} is the idealized estimator satisfying $\mathbb{E}[g_{\theta_k} \mid \mathcal{F}_k] = \nabla_\theta \mathcal{L}_D(\theta_k, \lambda_k)$ and $\mathbb{E}[\|g_{\theta_k}\|^2 \mid \mathcal{F}_k] \leq G^2$. By L-smoothness of $\mathcal{L}_D(\cdot, \lambda_k)$, the objective at θ_{k+1} satisfies:

$$\mathcal{L}_D(\theta_{k+1}, \lambda_k) \leq \mathcal{L}_D(\theta_k, \lambda_k) + \langle \nabla_\theta \mathcal{L}_D(\theta_k, \lambda_k), \theta_{k+1} - \theta_k \rangle + \frac{L}{2}\|\theta_{k+1} - \theta_k\|^2.$$

Substituting $\theta_{k+1} - \theta_k = -\eta_{\theta,k} g_{\theta_k}$ and taking the conditional expectation yields:

$$\mathbb{E}[\mathcal{L}_D(\theta_{k+1}, \lambda_k) \mid \mathcal{F}_k] \leq \mathcal{L}_D(\theta_k, \lambda_k) - \eta_{\theta,k}\|\nabla_\theta \mathcal{L}_D(\theta_k, \lambda_k)\|^2 + \frac{LG^2\eta_{\theta,k}^2}{2}.$$

Step 2: Bounding the Primal Gap via the PL Condition. Assume that $\mathcal{L}_D(\cdot, \lambda_k)$ satisfies the uniform μ-PL condition:

$$\frac{1}{2}\|\nabla_\theta \mathcal{L}_D(\theta, \lambda_k)\|^2 \geq \mu(\mathcal{L}_D(\theta, \lambda_k) - \mathcal{L}_D(\theta^*_{\lambda_k}, \lambda_k)),$$

where $\theta^*_{\lambda_k}$ is a minimizer over the fixed region for the current dual variable. Conditioning on $\mathcal{F}_k$ and subtracting the minimum gives:

$$\mathbb{E}[\mathcal{L}_D(\theta_{k+1}, \lambda_k) - \mathcal{L}_D(\theta^*_{\lambda_k}, \lambda_k) \mid \mathcal{F}_k] \leq (1 - 2\mu\eta_{\theta,k})(\mathcal{L}_D(\theta_k, \lambda_k) - \mathcal{L}_D(\theta^*_{\lambda_k}, \lambda_k)) + \frac{LG^2\eta_{\theta,k}^2}{2}.$$

Step 3: Quantifying the Shift in the Dual Penalty Landscape. Concurrently, the dual variable λ is updated on a slower timescale via gradient ascent:

$$\lambda_{k+1} = \max\{0, \lambda_k + \eta_{\lambda,k} \cdot (\mathcal{L}_A(\theta_k) - \epsilon)\}.$$

Let $m(\lambda)$ be the minimum of $\mathcal{L}_D(\cdot, \lambda)$ over this region and $\delta_k = \mathbb{E}[\mathcal{L}_D(\theta_k, \lambda_k) - m(\lambda_k)]$. Since $|c(\theta)| \leq B$ over the region, $|\mathcal{L}_D(\theta, \lambda') - \mathcal{L}_D(\theta, \lambda)| \leq B|\lambda' - \lambda|$. Taking minima gives $|m(\lambda') - m(\lambda)| \leq B|\lambda' - \lambda|$. Thus the primal gap at a fixed θ changes by at most $2B|\lambda' - \lambda|$. Using $|\lambda_{k+1} - \lambda_k| \leq B\eta_{\lambda,k}$ and taking expectations yields:

$$\delta_{k+1} \leq (1 - 2\mu\eta_{\theta,k})\delta_k + \frac{LG^2\eta_{\theta,k}^2}{2} + 2B^2\eta_{\lambda,k}.$$

Step 4: Sublinear Primal Tracking. Set $\eta_{\theta,k} = c/(k + K_0)$ with $2\mu c > 1$ and K_0 sufficiently large, and assume $\eta_{\lambda,k} \leq C_\lambda \eta_{\theta,k}^2$. The recurrence becomes $\delta_{k+1} \leq (1 - a/(k + K_0))\delta_k + d/(k + K_0)^2$ for $a = 2\mu c > 1$ and a finite constant d. Choosing $M \geq \max\{(1 + K_0)\delta_1, d/(a - 1)\}$ and applying induction gives $\delta_k \leq M/(k + K_0)$, hence $\delta_k = O(1/k)$, as stated in Theorem 5.2.

Table 3: Hyperparameter configuration across methods.

Hyperparameter	**SFT / OPD**	**ExOPD**	**LAMP**
Hardware	4×A100 (80GB)	4×A100 (80GB)	4×A100 (80GB)
Effective Batch Size	128	128	128
Training Epochs	3	3	3
Primal LR η_θ	5.0×10^{-5}	5.0×10^{-5}	5.0×10^{-5}
Rollout Temp / Top-p	0.8 / 0.95	0.8 / 0.95	0.8 / 0.95
LoRA r / α	16 / 32	16 / 32	16 / 32
Extrapolation α_{ex}	1.0	1.25	1.0
Initial λ_0	–	–	0.1
Dual LR η_λ	–	–	1.0×10^{-6}
Divergence Bound ϵ	–	–	0.15

C Detailed Experimental Setup

C.1 Base Models Description

Qwen2 Family. The Qwen2 series (Yang et al., 2024) employs SwiGLU activation functions, Grouped Query Attention (GQA), and Dual-Chunk Attention with YARN positional encoding for extended context handling. The models are pre-trained on high-quality multilingual corpora spanning code, mathematics, and general text. We use Qwen2-7B as the teacher model (7.07B parameters, 28 attention heads, 28 layers, hidden size 5632) and Qwen2-1.5B as the student (1.54B parameters, 12 attention heads, 28 layers, hidden size 3584).

Qwen2.5 Family. The Qwen2.5 series (Yang et al., 2025b) extends the Qwen2 architecture with expanded pre-training corpora (18 trillion tokens), enhanced data filtering pipelines, and refined tokenizer design. We employ Qwen2.5-7B as the teacher and Qwen2.5-1.5B as the student, utilizing parameter counts and architectural dimensions identical to their Qwen2 counterparts to ensure structural consistency across evaluations.

Gemma 2 Family. Gemma 2 (Team et al., 2024) introduces an interleaved attention mechanism that combines local sliding-window attention (window size 4096) with periodic global attention. This balances computational efficiency with long-range dependency modeling. We use Gemma2-9B (9.24B parameters, 42 layers, 16 attention heads, hidden size 3584) as the teacher and Gemma2-2B (2.61B parameters, 26 layers, 8 attention heads, hidden size 2304) as the student.

C.2 Dataset Description

Training Data. We construct a comprehensive 60k multi-domain training mixture from three curated sources:

- *UltraChat200k* (Ding et al., 2023): We randomly sample 20k instructional dialogue prompts spanning open-ended dialogue, knowledge-grounded QA, summarization, and creative writing to foster general instruction-following capabilities.
- *MetaMathQA* (Yu et al., 2024): We sample 20k mathematically focused prompts covering arithmetic, algebra, geometry, and complex word problems, constructed via answer augmentation and diverse-chain reasoning.
- *WizardCoder/Evol-Instruct* (Luo et al., 2024; Xu et al., 2023): We sample 20k code generation prompts spanning debugging, refactoring, and multi-language algorithmic problem-solving, generated through evolutionary instruction refinement.

Evaluation Data. Evaluation spans six rigorous benchmarks:

- *Instruction Following:* Evaluated on a held-out test set of 218 complex *Evol-Instruct* (Xu et al., 2023) prompts and the *UltraFeedback* (Cui et al., 2023) standard evaluation split.
- *Mathematical Reasoning:* Evaluated on *GSM8K* (Cobbe et al., 2021) (1.3K grade-school arithmetic test problems) and *MATH* (Hendrycks et al., 2021) (5K challenging competition-level test problems).

- *Code Generation:* Evaluated on *HumanEval* (Chen et al., 2021) (164 hand-written Python problems) and *MBPP* (Austin et al., 2021) (500 crowd-sourced Python problems). Both are assessed using the rigorous EvalPlus (Chen et al., 2021) test harness.

C.3 Baseline Details

All methods share a common two-phase training pipeline to ensure controlled, highly reproducible comparisons:

Phase 1: Supervised Fine-Tuning (SFT). The base student model is fine-tuned on the full 60k multi-domain mixture using standard next-token prediction with cross-entropy loss, producing the *Generalist SFT Checkpoint* shared by all downstream variants.

Phase 2: On-Policy Distillation Methods. Four variants are systematically evaluated:

- *OPD (Single-Teacher On-Policy Distillation):* The student generates rollouts scored by a single generalist teacher model (the 7B/9B model fine-tuned on the full 60k mixture).
- *MOPD (Multi-Teacher On-Policy Distillation):* Employs three domain-specialist teachers (fine-tuned exclusively on their respective 20k IF, Math, or Code subsets). Prompts are hard-routed to their corresponding specialist without additional regularization.
- *ExOPD (Extrapolated MOPD):* MOPD augmented with reward extrapolation. The implicit reward signal is extrapolated beyond the teacher's distribution using scaling coefficient $\alpha = 1.25$. The SFT checkpoint serves as the reference policy π_{ref}.
- *LAMP (Ours):* MOPD dynamically constrained via Lagrangian Anchoring. The static anchor model π_A is set to the frozen Generalist SFT Checkpoint. Parameters are set to $\lambda_0 = 0.1$, divergence bound $\epsilon = 0.15$, and dual learning rate $\eta_\lambda = 1 \times 10^{-6}$.

C.4 Training Details

All training is conducted on a node with 4×NVIDIA A100 (80GB) GPUs. To optimize memory, we employ Low-Rank Adaptation (LoRA) (Hu et al., 2022) with rank $r = 16$, $\alpha = 32$, and dropout set to 0.05, applied across all linear layers (Q, K, V, O projections, and MLP layers). The primal learning rate is 5×10^{-5} utilizing a cosine decay schedule with a 3% warmup period. Optimization runs for 3 epochs with an effective batch size of 128 (gradient accumulation across micro-batches of 4).

During on-policy updates, generation is accelerated via vLLM. Rollouts utilize a temperature of 0.8, nucleus sampling top-$p = 0.95$, and a maximum sequence length of 512 tokens. Teacher log-probabilities are pre-computed offline to mitigate GPU idling. For LAMP, the dual variable λ is updated at every optimization step based on the pre-update constraint violation $\mathcal{L}_A(\theta)$.

C.5 Evaluation Details

All downstream evaluations are executed on a single NVIDIA A100 (80GB) GPU.

Instruction Following. Responses are generated using temperature 0.8, top-$p = 0.95$, and a maximum length of 512 tokens. Winning Rate (%) is computed via a bidirectional LLM-as-a-Judge protocol using GPT-4o and GPT-4o-mini (evaluation temperature 0.7). Swapping prompt positions ensures the complete elimination of positional bias (Zheng et al., 2023). To improve reliability, we sample five responses per input using different random seeds ($\{10, 20, 30, 40, 50\}$) and report metrics averaged across these samples.

Mathematical Reasoning. We employ greedy decoding (temperature 0) with a maximum token length of 1024. Pass@1 is computed via an exact match evaluation after rigorous text normalization using rule-based parsers designed to isolate the final numerical output. Self-consistency or majority voting mechanisms are strictly omitted to evaluate raw model capability.

Code Generation. Greedy decoding (temperature 0) is employed with a maximum length of 1024 tokens. Functional correctness (Pass@1) is computed using the EvalPlus sandbox (Chen et al., 2021), which systematically executes generated code against both held-out and automatically mutated test cases, yielding a highly stringent evaluation of functional accuracy.

D LIMITATIONS

A primary limitation of the LAMP framework is the non-trivial computational and memory overhead introduced by the dynamic anchoring mechanism during the post-training phase. At each optimization step, computing the dual advantage requires tracking token-level log-probabilities across multiple distinct entities: the active student policy π_θ, the targeted domain specialist teacher π_T, and the static anchor reference model π_A. Consequently, executing concurrent forward passes across these large-scale architectures substantially increases the overall compute requirements and GPU memory footprint per training iteration when compared to standard single-teacher or unconstrained multi-teacher distillation baselines.