COACH: ASYMMETRIC ANCHOR–HELPER ROUTING IN MIXTURES OF LORA EXPERTS

Anonymous authors
Paper under double-blind review

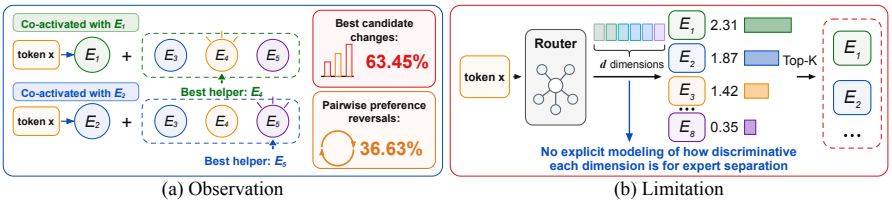
ABSTRACT

Combining Low-Rank Adaptation (LoRA) with Mixture-of-Experts (MoE) has been widely studied for parameter-efficient multi-task adaptation of large language models (LLMs). However, the existing routing methods have not fully explored dimension-wise discriminative information across expert regions or candidate utility conditioned on selected experts, potentially limiting multi-task adaptation performance. To address this limitation, this paper proposes **COACH**, an asymmetric Anchor–Helper routing approach for mixtures of LoRA experts based on **C**onditioning **o**n **A**nchors for **C**alibrating **H**elpers. Specifically, COACH comprises two complementary modules: Prototype-Guided Dimension Reweighting (PDR) exploits dimension-wise expert discriminability to stabilize base matching, while Cluster Residual Collaboration (CRC) uses an Anchor-conditioned residual-response prior to recalibrate Helper selection, addressing the lack of selected-expert conditioning in one-shot routing. Together, these two modules calibrate routing without introducing additional trainable parameters or auxiliary training objectives, while preserving the prescribed sparse activation budget. From a feature-weighted clustering perspective, the theoretical analysis derives the optimal dimension weights for PDR, proves that more discriminative dimensions receive higher weights, and applies this result to calibrate base routing. Extensive experiments across seven tasks show that COACH achieves optimal performance in both macro accuracy (equally averaged over tasks) and micro accuracy (computed over all evaluation examples) among the compared methods, exceeding the equally weighted mean performance of six external baselines by **1.87** and **2.11** percentage points, respectively. Further mechanistic experiments demonstrate that PDR improves routing separation and perturbation stability, while CRC selects more useful Helpers through Anchor-conditioned calibration.

1 INTRODUCTION

Recent studies have explored diverse routing strategies for mixtures of LoRA experts, including approaches that improve input–expert matching and those that model interactions among experts. For input–expert matching, MoLE (Wu et al., 2024) learns layer-wise gates for LoRA composition, while AdaMoLE (Liu & Luo, 2024) adapts expert activation through input-dependent thresholds. Beyond the gating rule, routing objectives help shape expert allocation: HMoRA (Liao et al., 2025) balances expert utilization and selection certainty through divergence-based regularization, while LoRA-Mixer (Li et al., 2026) promotes input-aware specialization through a routing specialization objective. Semantic information provides additional guidance, with SAMoRA (Shi et al., 2026) aligning input semantics with expert capabilities and TopicLoRA (Xu et al., 2026) guiding expert allocation through topic-based priors. At the feature level, the Adaptive Clustering (AC) router (Nielsen et al., 2025) in general MoE models reweights input dimensions according to within-cluster dispersion. Beyond matching inputs to individual experts, another line of work explicitly models expert interactions. At the weighting level, TeamLoRA (Lin et al., 2025) adjusts expert weights through a learnable interaction matrix. Moving from weights to representations, TalkLoRA (Mu et al., 2026) enables communication among low-rank expert representations before routing. To account for layer-wise differences, HotMoE (Huang et al., 2026) refines routing scores through expert–expert attention in lower layers and token–expert attention in higher layers. Extending interaction across computation rounds, GraphMoE (Lv et al., 2026b) uses recurrent routing to propagate information among experts.

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.



(a) Observation (b) Limitation

Figure 1: **Observation and limitation.** (a) Observation of expert utility under different expert pairings. (b) Limitation of conventional Top-K routing.

However, despite the progress made by prior work, the following issues remain and the performance still has room for further improvement. First, the existing routing does not explicitly exploit dimension-wise discriminative differences across expert input regions to calibrate base matching. Second, one-shot Top-K selection does not re-evaluate remaining candidates conditioned on already selected experts, despite the potential dependence of candidate utility on expert composition.

Meanwhile, controlled experiments demonstrate that candidate preferences vary with the co-activated expert. As shown in Figure 1 (a), changing only this expert while fixing the input, candidate set, and routing weights changes the best candidate in 63.45% of routing events and reverses the ordering of 36.63% of comparable candidate pairs. These observations motivate conditioning candidate evaluation on an already selected expert, rather than selecting the entire set at once as illustrated in Figure 1 (b).

To address these limitations, this paper proposes COACH, comprising two modules: Prototype-Guided Dimension Reweighting (PDR) and Cluster Residual Collaboration (CRC). PDR reweights routing dimensions based on cross-expert prototype differences to select the primary expert. CRC then uses candidate experts' residual responses to the Anchor prototype to guide the selection of additional experts.

The main contributions are summarized as follows:

• This paper proposes an asymmetric Anchor–Helper routing approach, **COACH**, for mixtures of LoRA experts, comprising Prototype-Guided Dimension Reweighting (**PDR**) and Cluster Residual Collaboration (**CRC**). PDR refines the initial input–expert correspondence through prototype-informed feature weighting, whereas CRC introduces an Anchor-dependent adjustment to identify complementary Helpers.

• Theoretical analysis from a feature-weighted clustering perspective proves that, under a reverse-KL-regularized prototype-separation objective, optimal dimension-wise weights increase monotonically with expert discriminability: dimensions that better distinguish expert input regions receive higher weights. This result motivates **PDR**, which calibrates base routing by emphasizing more discriminative dimensions.

• To account for composition-dependent candidate utility, a conditional calibration module, **CRC**, is developed to guide Helper selection by constructing Anchor-conditioned residual-response priors for the remaining candidates and using these priors to recalibrate their selection scores after the Anchor is fixed.

• Extensive experiments across seven tasks demonstrate the effectiveness of **COACH**, which achieves state-of-the-art aggregate macro- and micro-averaged performance among the compared methods. Ablation studies and controlled analyses further validate the complementary contributions of **PDR** and **CRC**, with PDR consistently improving routing separation and robustness, and CRC yielding more effective Helper choices under fixed Anchors and mixture weights.

2 Related Work

2.1 Multi-Task Low-Rank Adaptation

Low-Rank Adaptation (LoRA) (Hu et al., 2022) enables parameter-efficient adaptation of large language models by learning low-rank updates while keeping pretrained weights frozen. For multi-task

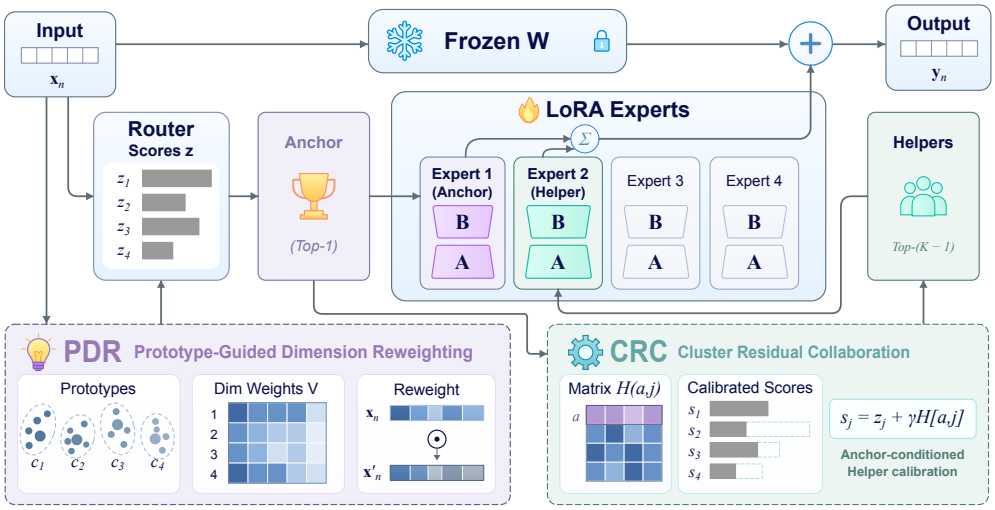


Figure 2: **Overview of the proposed COACH.** PDR reweights routing dimensions using expert prototypes to determine the Anchor, while CRC conditions on the selected Anchor to recalibrate candidate scores for Helper selection. Together, they improve base matching and multi-expert selection under the same sparse activation budget.

adaptation, existing studies explore the composition, sharing, and construction of LoRA experts. LoRAHub (Huang et al., 2024) and mtLoRA (Tian et al., 2026) investigate cross-task LoRA composition and structured multi-task adaptation, respectively. MoLoRA (Zadouri et al., 2024), MOELoRA (Liu et al., 2024), LoRAMoE (Dou et al., 2024), MoLE (Wu et al., 2024), and MeteoRA (Xu et al., 2025) integrate LoRA with Mixture-of-Experts (MoE), combining multiple LoRA experts through different gating mechanisms. In addition, HydraLoRA (Tian et al., 2024), MoDE (Ning et al., 2025), LoRACoE (Li et al., 2025), and D2MoRA (Zuo et al., 2026) explore alternative expert architectures through parameter sharing, rank-level composition, and expert decomposition.

2.2 Expert Routing in MoE-LoRA

Beyond expert architecture, existing studies improve LoRA expert routing and selection from several perspectives. AdaMoLE (Liu & Luo, 2024) and LD-MoLE (Zhuang et al., 2026) dynamically adjust the number of activated experts. HMoRA (Liao et al., 2025) combines token-level and task-level signals through hierarchical routing, while SAMoRA (Shi et al., 2026) and TopicLoRA (Xu et al., 2026) incorporate semantic information or topic priors to guide expert allocation. Other studies focus on expert specialization and interactions: LoRA-Mixer (Li et al., 2026) promotes input-aware expert specialization through routing optimization, TalkLoRA (Mu et al., 2026) enables communication among low-rank expert representations, and HotMoE (Huang et al., 2026) incorporates expert–expert and token–expert relationships into routing scores. While prior work incorporates additional information or expert interactions into routing, COACH adopts an asymmetric routing approach that improves routing separation and stability while enabling more effective Helper selection.

3 Method

This paper proposes COACH, an asymmetric Anchor–Helper routing approach for mixtures of LoRA experts, to address the limited use of dimension-wise expert discriminability in base matching and the lack of selected-expert conditioning in multi-expert selection. As illustrated in Figure 2, PDR calibrates base routing scores to determine the primary expert (Anchor), while CRC subsequently uses Anchor-conditioned residual responses to select additional experts (Helpers).

Let $\mathbf{x}_n^{(l)} \in \mathbb{R}^d$ denote the input representation of token n at a routed module in layer l, where d is the hidden dimension. Each token activates K of E experts.

3.1 PROTOTYPE-GUIDED DIMENSION REWEIGHTING (PDR)

Hidden dimensions differ in their ability to distinguish expert input regions. PDR uses running expert prototypes to construct expert-specific dimension weights for base routing. Each expert e maintains a non-gradient prototype $\boldsymbol{\mu}_e^{(l)} \in \mathbb{R}^d$, with details provided in Appendix B.1. The discriminability score of dimension q for expert e is defined as

$$\sigma_{e,q}^{(l)} = \sum_{j \neq e} \left(\mu_{e,q}^{(l)} - \mu_{j,q}^{(l)} \right)^2 . \tag{1}$$

Let $\sigma_{e,\max}^{(l)}$ and $\bar{\sigma}_e^{(l)}$ denote the maximum and mean discriminability scores for expert e. PDR constructs normalized dimension weights $\mathbf{p}_e^{(l)}$ and routing reweighting vectors $\mathbf{v}_e^{(l)}$ as

$$\begin{aligned} \alpha_e^{(l)} &= \sigma_{e,\max}^{(l)} + \delta_\alpha \left(\sigma_{e,\max}^{(l)} - \bar{\sigma}_e^{(l)} \right) + 2\epsilon, \\ p_{e,q}^{(l)} &= \frac{\left(\alpha_e^{(l)} - \sigma_{e,q}^{(l)} \right)^{-1}}{\sum_{q'=1}^{d} \left(\alpha_e^{(l)} - \sigma_{e,q'}^{(l)} \right)^{-1}}, \\ \mathbf{v}_e^{(l)} &= \mathbf{1} + \rho_l \left(d\mathbf{p}_e^{(l)} - \mathbf{1} \right), \end{aligned} \tag{2}$$

where $\delta_\alpha \geq 0$ controls weight concentration, $\epsilon > 0$ ensures positive denominators, and $\mathbf{1}$ is the all-ones vector. The fixed coefficient $\rho_l \in [0, 1]$ controls interpolation with identity weighting and increases with layer depth.

After the warm-up interpolation and numerical clipping specified in Appendix B.1, the effective weights $\mathbf{v}_e^{(l),\text{eff}}$ determine the base routing logits:

$$z_{n,e}^{(l)} = \left(\mathbf{w}_e^{(l)} \right)^\top \left(\mathbf{v}_e^{(l),\text{eff}} \odot \mathbf{x}_n^{(l)} \right) + b_e^{(l)}, \tag{3}$$

where $\mathbf{w}_e^{(l)}$ and $b_e^{(l)}$ are the router weight vector and bias, respectively, and $\odot$ denotes element-wise multiplication. Reweighting affects only routing; LoRA experts receive the original input $\mathbf{x}_n^{(l)}$.

PDR therefore turns cross-expert prototype differences into a discriminability-aware routing metric, focusing base matching on dimensions that better distinguish expert regions and sharpening the separation among competing experts.

3.2 CLUSTER RESIDUAL COLLABORATION (CRC)

CRC supplements token-specific base scores with a regional prior for Helper selection. The Anchor is first fixed as $a_n^{(l)} = \arg\max_e z_{n,e}^{(l)}$. For an Anchor candidate i and a Helper candidate j, CRC constructs the response matrix $\mathbf{C}^{(l)}$ through

$$C_{i,j}^{(l)} = \left\| s_{\text{L}} \mathbf{B}_j^{(l)} \mathbf{A}_j^{(l)} \boldsymbol{\mu}_i^{(l)} \right\|_2 , \tag{4}$$

where $\mathbf{A}_j^{(l)}$ and $\mathbf{B}_j^{(l)}$ are the LoRA matrices of expert j, and s_{L} is the LoRA scaling factor. The response magnitude serves as a region-conditioned proxy for candidate selection.

Log transformation and row-wise standardization of the off-diagonal entries produce a conditional prior $\mathbf{H}^{(l)} \in \mathbb{R}^{E \times E}$, with $H_{i,i}^{(l)} = 0$. This normalization emphasizes relative candidate responses within each region. The prior is periodically updated without gradients, with normalization and numerical details provided in Appendix B.1.

CRC calibrates only non-Anchor candidates:

$$s_{n,e}^{(l)} = \begin{cases} z_{n,e}^{(l)} + \gamma_t H_{a_n^{(l)},e}^{(l)}, & e \neq a_n^{(l)}, \\ z_{n,e}^{(l)}, & e = a_n^{(l)}, \end{cases} \tag{5}$$

where γ_t is the prior strength at step t, increased to γ after warm-up as detailed in Appendix B.1. The active set retains the Anchor and selects $K - 1$ Helpers from the rest as

$$\mathcal{S}_n^{(l)} = \left\{a_n^{(l)}\right\} \cup \mathrm{Top}_{K-1}\left(\left\{s_{n,e}^{(l)} : e \neq a_n^{(l)}\right\}\right), \tag{6}$$

where Top_{K-1} returns the indices of the $K-1$ highest-scoring eligible experts. Thus, the regional prior adjusts Helper selection without changing the Anchor identity.

Finally, the calibrated scores are normalized within the active set to aggregate the selected expert outputs. COACH uses only the standard task loss with label smoothing, with the mixture weights, module output, and training objective given by

$$\begin{aligned} g_{n,e}^{(l)} &= \frac{\exp\left(s_{n,e}^{(l)}/\tau_l\right)}{\sum_{j\in\mathcal{S}_n^{(l)}} \exp\left(s_{n,j}^{(l)}/\tau_l\right)}, \qquad e \in \mathcal{S}_n^{(l)}, \\ \mathbf{y}_n^{(l)} &= \mathbf{W}^{(l)}\mathbf{x}_n^{(l)} + s_{\mathrm{L}} \sum_{e\in\mathcal{S}_n^{(l)}} g_{n,e}^{(l)} \mathbf{B}_e^{(l)} \mathbf{A}_e^{(l)} \mathbf{x}_n^{(l)}, \\ \mathcal{L}_{\mathrm{COACH}} &= \mathcal{L}_{\mathrm{task}} = -\frac{1}{N}\sum_{n=1}^{N} \log p_\theta(t_n \mid x_n). \end{aligned} \tag{7}$$

Here, t_n denotes the ground-truth target and $p_\theta(t_n \mid x_n)$ is its predicted probability. $\tau_l > 0$ is the layer-dependent routing temperature, $\mathbf{W}^{(l)}$ is the frozen pretrained projection, and mixture weights outside the active set are zero. PDR and CRC introduce no additional trainable parameters or auxiliary objectives, only router and LoRA parameters are optimized.

CRC therefore makes Helper selection explicitly dependent on the selected Anchor, allowing candidate rankings to adapt to the expert composition rather than remain fixed by token–expert matching.

4 Theoretical Analysis

This paper analyzes PDR's prototype-separation and dimension-allocation properties for fixed running prototypes and pre-clipping weights through a reverse-KL-regularized feature-weighting formulation based on cross-expert prototype separation. Fix an expert and layer, and let $\boldsymbol{\sigma}$ denote the scores in Eq. (1). For weights $\mathbf{p}$ in the positive probability simplex Δ_d°, define $G(\mathbf{p}) = \boldsymbol{\sigma}^\top \mathbf{p}$, the weighted squared distance from this expert's prototype to the others.

For nonconstant scores, maximizing $G(\mathbf{p})$ alone drives weight concentration toward the most discriminative dimensions at the simplex boundary. Therefore, consider the reverse-KL-regularized prototype-separation problem

$$\max_{\mathbf{p}\in\Delta_d^\circ} \left\{G(\mathbf{p}) - \lambda D_{\mathrm{KL}}(\mathbf{u}\|\mathbf{p})\right\}, \qquad \lambda > 0, \quad \mathbf{u} = \mathbf{1}/d. \tag{8}$$

The reverse-KL term prevents degenerate concentration while allowing larger weights to be assigned to dimensions with stronger prototype discriminability.

Lemma 1 (Optimal weighting for regularized prototype separation)**.** *For finite discriminability scores $\boldsymbol{\sigma}$ and $\lambda > 0$, the objective in Eq. (8) has a unique maximizer $\mathbf{p}^\star$. There exists a unique $\alpha > \sigma_{\max}$, where $\sigma_{\max} = \max_q \sigma_q$, such that*

$$p_q^\star = \frac{1}{Z(\alpha - \sigma_q)}, \qquad Z = \sum_{r=1}^{d} \frac{1}{\alpha - \sigma_r}, \qquad \lambda = \frac{d}{Z}. \tag{9}$$

Proof. The complete derivation and uniqueness proof are provided in Appendix A.1. □

Theorem 1 (Prototype-separation enhancement under PDR)**.** *The practical PDR parameterization in Eq. (2) defines*

$$\alpha = \sigma_{\max} + \delta_\alpha(\sigma_{\max} - \bar{\sigma}) + 2\epsilon,$$

and therefore

$$\alpha - \sigma_q = (\sigma_{\max} - \sigma_q) + \delta_\alpha(\sigma_{\max} - \bar{\sigma}) + 2\epsilon > 0. \tag{10}$$

Hence, the normalized dimension weights in Eq. (2) coincide with the solution $\mathbf{p}^\star$ in Lemma 1 for the induced regularization strength $\lambda_{\mathrm{eff}} = d/Z$. For any $\rho_l \in [0,1]$, let $\bar{\mathbf{p}} = (1-\rho_l)\mathbf{u} + \rho_l \mathbf{p}^\star$. Then

$$G(\bar{\mathbf{p}}) - G(\mathbf{u}) \geq \rho_l \lambda_{\mathrm{eff}} D_{\mathrm{KL}}(\mathbf{u}\|\mathbf{p}^\star) \geq 0. \tag{11}$$

The gain is strictly positive if and only if $\rho_l > 0$ and the discriminability scores are not all equal.

Table 1: **Multi-task performance on Qwen2-1.5B.** TP denotes trainable parameters in millions. Results are mean test accuracies (%) over shared random seeds, sample standard deviations appear below. Macro-Avg averages tasks equally, while Micro-Avg weights tasks by test-set size. Best and second-best results are bolded and underlined, respectively.

Method	TP (M)	QASC	DREAM	Cosmos QA	MuTual	Wino Grande	ART	ANLI	Macro-Avg	Micro-Avg
LoRA	1.09	73.15 ±0.76	85.15 ±0.58	76.14 ±0.97	83.22 ±0.28	65.32 ±0.48	77.35 ±0.11	48.92 ±0.61	72.75 ±0.25	71.85 ±0.51
MoLoRA	77.93	73.61 ±2.22	85.95 ±0.61	82.27 ±1.22	87.10 ±0.77	73.22 ±0.24	79.33 ±0.61	53.08 ±0.87	76.37 ±0.16	76.18 ±0.45
TalkLoRA	13.99	<u>74.02</u> ±0.00	<u>86.13</u> ±0.26	78.76 ±0.27	<u>87.36</u> ±0.60	68.35 ±1.03	78.11 ±0.32	50.84 ±0.34	74.80 ±0.18	73.90 ±0.12
HMoRA	60.81	73.97 ±1.50	85.20 ±0.44	79.21 ±0.72	85.48 ±0.07	67.93 ±0.43	78.37 ±0.26	51.66 ±1.62	74.54 ±0.30	74.02 ±0.28
HotMoE	49.23	72.28 ±0.59	85.49 ±0.05	<u>82.40</u> ±0.14	86.46 ±0.30	71.80 ±0.53	78.96 ±0.94	52.89 ±0.97	75.75 ±0.22	75.86 ±0.33
mtLoRA	13.07	72.79 ±1.59	86.09 ±0.50	82.16 ±0.31	<u>87.36</u> ±0.11	<u>73.69</u> ±0.83	<u>79.63</u> ±0.52	**53.54** ±0.51	<u>76.47</u> ±0.10	<u>76.30</u> ±0.08
COACH	25.86	**74.12** ±0.53	**86.44** ±0.97	**82.91** ±0.74	**87.66** ±0.88	**74.40** ±1.06	**80.05** ±0.10	<u>53.30</u> ±0.55	**76.98** ±0.16	**76.79** ±0.22

Proof. The complete proof and exact gain decomposition are provided in Appendix A.2. □

Thus, before clipping, PDR preserves or improves aggregate prototype separation relative to uniform dimension weighting under the same total weight budget.

The closed-form solution also makes the dimension-allocation rule explicit. From the optimal solution in Eq. (9),

$$\sigma_q > \sigma_r \quad \Longrightarrow \quad p_q^\star > p_r^\star, \qquad \sigma_q = \sigma_r \quad \Longrightarrow \quad p_q^\star = p_r^\star. \tag{12}$$

Thus, the optimal weighting assigns larger weights to dimensions with stronger prototype discriminability and equal weights to equally discriminative dimensions, formally characterizing the allocation rule implemented by PDR. For the pre-clipping routing weights $\mathbf{v} = d\bar{\mathbf{p}}$, the same strict ordering is retained when $\rho_l > 0$. If all scores coincide, $\mathbf{p}^\star = \mathbf{u}$ and $\mathbf{v} = \mathbf{1}$. The metric correspondence and allocation properties are detailed in Appendix A.2. Effects on learned routing margins, Anchor stability, and downstream performance are evaluated empirically in Section 5.3.

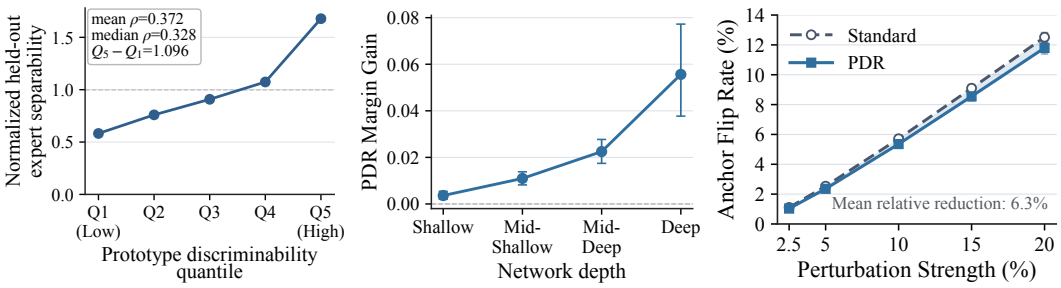
5 Experiments

5.1 Experimental Setup

Benchmarks and Metrics. COACH is evaluated on seven tasks: QASC (Khot et al., 2020), DREAM (Sun et al., 2019), CosmosQA (Huang et al., 2019), MuTual (Cui et al., 2020), WinoGrande (Sakaguchi et al., 2020), ART (Bhagavatula et al., 2020), and ANLI (Nie et al., 2020), covering scientific and commonsense reasoning, reading comprehension, dialogue understanding, and natural language inference. Joint training uses task-balanced sampling. Main and ablation comparisons use independent test splits, reporting task-wise Accuracy, Macro Accuracy averaged equally over tasks, and Micro Accuracy over test examples.

Baselines. Using Qwen2-1.5B Base (Yang et al., 2024), COACH is compared with LoRA (Hu et al., 2022), MoLoRA (Zadouri et al., 2024), TalkLoRA (Mu et al., 2026), HMoRA (Liao et al., 2025), mtLoRA (Tian et al., 2026), and HotMoE (Huang et al., 2026). All methods share the training data, evaluation protocol, and budget of 10,000 optimizer updates; LoRA rank and expert budgets are matched where compatible with their designs. Results are averaged over multiple shared random seeds. Method-specific configurations are provided in the Appendix.

Implementation. COACH uses $E = 8$ LoRA experts with Top-2 routing in the Attention q_{proj}, k_{proj}, v_{proj}, and o_{proj} modules. These experts use rank $r = 8$ and LoRA scaling parameter $\alpha = 16$.



(a) Prototype discriminability and expert separability.

(b) Routing margin across layers.

(c) Anchor stability under perturbations.

Figure 3: **Mechanistic analysis of PDR.** (a) Held-out expert separability across prototype-discriminability groups. (b) Normalized routing-margin gains from PDR across depth groups. (c) Anchor flip rates under paired perturbations.

Defaults are $\delta_\alpha = 1.0$ and $\gamma = 0.25$. COACH is optimized with AdamW (Loshchilov & Hutter, 2019) for 10,000 steps at effective batch size 32 in BF16 precision, without additional trainable parameters or auxiliary objectives. Optimization and statistical-update details are provided in the Appendix.

5.2 Overall Performance

The overall performance of COACH is summarized in Table 1. Across the seven benchmarks, COACH achieves the optimal aggregate performance, with **76.98% Macro-Avg** and **76.79% Micro-Avg**. Compared with the mean performance across all baselines, COACH improves Macro-Avg and Micro-Avg by **1.87** and **2.11** percentage points, respectively. COACH further achieves (near-) optimal performance across all seven benchmarks, ranking first on QASC, DREAM, CosmosQA, MuTual, WinoGrande, and ART, and second on ANLI. These results demonstrate consistent improvements across scientific and commonsense reasoning, reading comprehension, dialogue understanding, and natural language inference.

Table 1 also reports trainable parameters (TP) to compare performance and adaptation cost. COACH achieves the strongest aggregate performance with 25.86M parameters, fewer than several multi-expert baselines. Notably, PDR and CRC introduce no additional trainable parameters or auxiliary objectives beyond the LoRA expert configuration. To further assess COACH under different sparse routing budgets, experiments with varying Top-K activation settings are provided in the Appendix.

To better understand the source of these gains, the following mechanism analyses investigate how PDR and CRC individually contribute to COACH and how each module affects the routing process.

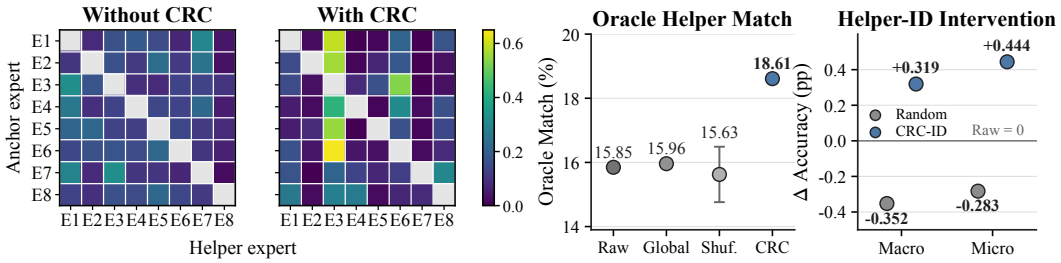
5.3 Mechanistic Analysis

5.3.1 Understanding PDR: Routing Separation and Robustness

The PDR analysis examines prototype discriminability, routing separation, and Anchor stability under perturbations. In Figure 3 (a), held-out expert separability increases with prototype discriminability, with a mean Spearman correlation of 0.372, supporting prototype discriminability as a signal for PDR reweighting. The definitions of Q_1–Q_5 and analysis details are provided in Appendix C.3.

Using the same frozen checkpoint, the routing-margin comparison retains only events where both methods select the same Anchor on the clean input. PDR increases the mean normalized Top-1/Top-2 margin from 0.3333 to 0.3557, strengthening routing separation. Figure 3 (b) shows larger gains in deeper layers. Implementation details and complete results are provided in Appendix C.3.

Finally, paired Gaussian perturbations assess Anchor stability on these matched events. As shown in Figure 3 (c), PDR reduces the Anchor flip rate across all tested perturbation strengths, with a mean relative reduction of 6.3%, supporting more stable Anchor selection. Here, Standard denotes routing without PDR. The perturbation protocol and detailed results are provided in Appendix C.3.



(a) Anchor-Conditioned Helper Preferences **(b) Helper Selection Quality and Utility**

Figure 4: **Mechanistic analysis of CRC.** (a) Anchor–Helper selection distributions with and without CRC. (b) Oracle Helper match rates and Helper-identity intervention results.

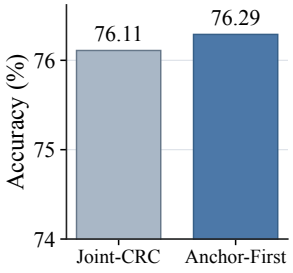
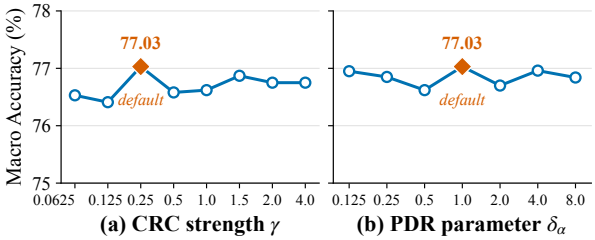


Figure 5: **Anchor-First versus Joint-CRC.** Macro Accuracy after separate training.



(a) CRC strength γ **(b) PDR parameter δ_α**

Figure 6: **Hyperparameter sensitivity of COACH.** (a) CRC strength (γ). (b) PDR parameter (δ_α).

5.3.2 Understanding CRC: Anchor-Conditioned Helper Selection

After the Anchor is determined, CRC uses the Anchor-specific collaboration prior to recalibrate the remaining candidate experts. The analysis examines CRC from three perspectives: whether it induces Anchor-dependent Helper preferences, whether those preferences improve Oracle Helper selection and downstream utility, and why the collaboration prior should be applied after the Anchor is established. The results are summarized in Figures 4 and 5.

To examine how Anchor-conditioned Helper calibration changes the resulting expert selection structure, Figure 4 (a) compares the empirical conditional distributions $P(H = j \mid A = i)$ of a separately trained Vanilla LoRA-MoE model ("Without CRC") and the full COACH model ("With CRC"), where A and H denote the selected Anchor and Helper expert indices, respectively. Compared with vanilla routing, COACH exhibits more differentiated Helper distributions across Anchors, indicating a shift toward more Anchor-specific expert pairings. We next examine whether these altered Helper selections correspond to higher-quality candidate choices and downstream utility.

To further test whether the Anchor-dependent structure provides useful conditional information beyond a global Helper preference, the left panel of Figure 4 (b) evaluates Oracle Helper matching over 3,584 routing events from 224 task-balanced validation examples. With the input, Anchor, and original routing weights fixed, all seven non-Anchor Helpers are enumerated, and the Oracle Helper is defined as the candidate minimizing the correct-option NLL. We compare **Raw**, which applies no CRC calibration; **Global**, which removes Anchor conditioning by using an Anchor-independent prior; **Shuffled**, which mismatches Anchors with prior rows; and **CRC**, which uses the prior conditioned on the actual Anchor. Further construction details are provided in Appendix C.4. CRC matches the Oracle Helper in 18.61% of routing events, compared with 15.85%, 15.96%, and 15.63% for Raw, Global, and Shuffled, respectively. Paired sample-clustered bootstrap gains over the three controls are $+2.762$, $+2.651$, and $+2.985$ pp, with all 95% confidence intervals above zero. These results indicate that the gains cannot be explained solely by global Helper preference or prior structure; correct Anchor conditioning provides useful information for Helper selection.

Table 2: **Component ablation of COACH on Qwen2-1.5B.** Base disables both PDR and CRC. Mean test accuracies (%) are averaged over shared random seeds; sample standard deviations appear below. Macro-Avg averages tasks equally, while Micro-Avg weights tasks by test-set size. Best and second-best mean accuracies are bolded and underlined, respectively.

Method	QASC	DREAM	Cosmos QA	MuTual	Wino Grande	ART	ANLI	Macro-Avg	Micro-Avg
Base	71.31 ±1.15	84.49 ±0.96	79.32 ±0.12	85.14 ±0.52	68.19 ±0.44	78.09 ±0.70	50.63 ±0.65	73.88 ±0.13	73.64 ±0.02
PDR-only	71.92 ±1.33	<u>85.26</u> ±0.61	81.99 ±0.18	<u>86.46</u> ±0.52	<u>71.67</u> ±1.12	<u>79.24</u> ±0.49	**53.43** ±0.65	<u>75.71</u> ±0.21	<u>75.79</u> ±0.04
CRC-only	<u>73.00</u> ±0.82	<u>85.26</u> ±0.32	<u>82.18</u> ±0.29	86.34 ±0.88	70.67 ±1.16	79.05 ±0.91	52.89 ±0.19	75.63 ±0.14	75.69 ±0.15
COACH	**74.12** ±0.53	**86.44** ±0.97	**82.91** ±0.74	**87.66** ±0.88	**74.40** ±1.06	**80.05** ±0.10	<u>53.30</u> ±0.55	**76.98** ±0.16	**76.79** ±0.22

To further assess whether improved Helper selection translates into downstream utility, the right panel of Figure 4 (b) performs a controlled Helper-identity intervention, fixing the Anchor and original routing weights while changing only the Helper identity. Replacing the Raw Helper with the CRC-selected Helper ("CRC-ID") improves Macro and Micro Accuracy by $+0.319$ pp and $+0.444$ pp, respectively, whereas random replacement ("Random") decreases both. This controlled comparison supports that CRC improves performance by selecting more useful Helpers, rather than through changes in the Anchor or mixture weights. Detailed intervention protocols are provided in Appendix C.4.

Finally, Joint-CRC tests whether the prior should also influence the Anchor. Anchor-First fixes the Anchor using token-specific logits before calibrating Helper selection; Joint-CRC scores candidate pairs using both logits and the prior, then selects the complete Top-2 set. Both models are trained separately under their respective policies from the start, allowing routers and LoRA experts to adapt. The formulation and training settings are provided in Appendix C.4. In Figure 5, Anchor-First achieves 76.29% Macro Accuracy, compared with 76.11% for Joint-CRC, with each model evaluated using its own learned experts, logits, and prior. These results support preserving the token-specific Anchor and applying CRC to Helper selection in the evaluated setting.

5.4 Ablation Study

Table 2 compares Base (both components disabled), PDR-only, CRC-only, and full COACH under matched trainable-parameter counts. Either component improves aggregate performance, and their combination performs best overall, with full COACH outperforming PDR-only on six of seven tasks and CRC-only on all seven. The slight drop relative to PDR-only on ANLI may arise from a task-specific interaction between the two calibrations: after PDR has refined base routing, CRC may occasionally alter otherwise effective Helper rankings. Figure 6 evaluates sensitivity to the CRC prior strength γ and PDR weight-concentration parameter δ_α, with the default value achieving the highest Macro-Avg in each sweep. Detailed experimental settings and results are provided in Appendix C.1.

6 Conclusion

This paper presents **COACH** for base matching and multi-expert selection in MoE-LoRA routing. PDR reweights base routing using dimension-wise expert differences to improve separation and stability, while CRC conditions Helper selection on the chosen Anchor using candidates' residual responses. Experiments across seven tasks show that COACH achieves the optimal average performance among the compared methods, with further analyses supporting the distinct roles of PDR and CRC. These results suggest that effective Top-K routing benefits from both dimension-wise expert discriminability and conditioning later selections on earlier ones. Future work may explore more general conditional routing and larger-scale settings.

AI USE STATEMENT

Generative AI tools, including large language models (LLMs), were used during the preparation of this manuscript to assist with language refinement and writing polish, including improving clarity, fluency, and the overall presentation of the text.

Generative AI tools were also used to support the retrieval and identification of relevant literature during the research process, helping to locate potentially related studies and materials for further review. In addition, they were used to assist with code completion and implementation, including supporting the development, refinement, and debugging of research code.

REFERENCES

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Scott Wen-tau Yih, and Yejin Choi. Abductive commonsense reasoning. In *International Conference on Learning Representations*, 2020.

Leyang Cui, Yu Wu, Shujie Liu, Yue Zhang, and Ming Zhou. MuTual: A dataset for multi-turn dialogue reasoning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 1406–1416, 2020.

Shihan Dou, Enyu Zhou, Yan Liu, Songyang Gao, Wei Shen, Limao Xiong, Yuhao Zhou, Xiao Wang, Zhiheng Xi, Xiaoran Fan, Shiliang Pu, Jiang Zhu, Rui Zheng, Tao Gui, Qi Zhang, and Xuanjing Huang. LoRAMoE: Alleviating world knowledge forgetting in large language models via MoE-style plugin. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pp. 1932–1945, 2024.

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

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. LoraHub: Efficient cross-task generalization via dynamic LoRA composition. In *First Conference on Language Modeling*, 2024.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 2391–2401, 2019.

Yitong Huang, Ziqi Yang, Zihui Wang, Jianzhong Qi, Rongshan Yu, Xiaoliang Fan, and Cheng Wang. Hybrid routing for a mixture of LoRA experts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 31211–31219, 2026.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. QASC: A dataset for question answering via sentence composition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 8082–8090, 2020.

Guanyu Li, Zhiheng Xi, Zhihao Zhang, Boyang Hong, Tao Gui, Qi Zhang, and Xuanjing Huang. LoRACoE: Improving large language model via composition-based LoRA expert. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 31290–31304, 2025.

Wenbing Li, Zikai Song, Hang Zhou, Junqing Yu, Yunyao Zhang, and Wei Yang. LoRA-Mixer: Coordinate modular LoRA experts through serial attention routing. In *International Conference on Learning Representations*, pp. 14694–14716, 2026.

Mengqi Liao, Wei Chen, Junfeng Shen, Shengnan Guo, and Huaiyu Wan. HMoRA: Making LLMs more effective with hierarchical mixture of LoRA experts. In *International Conference on Learning Representations*, pp. 91309–91330, 2025.

Tianwei Lin, Jiang Liu, Wenqiao Zhang, Yang Dai, Haoyuan Li, Zhelun Yu, Wanggui He, Juncheng Li, Jiannan Guo, Hao Jiang, Siliang Tang, and Yueting Zhuang. TeamLoRA: Boosting low-rank adaptation with expert collaboration and competition. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, pp. 13622–13637, 2025.

Qidong Liu, Xian Wu, Xiangyu Zhao, Yuanshao Zhu, Derong Xu, Feng Tian, and Yefeng Zheng. When MOE meets LLMs: Parameter efficient fine-tuning for multi-task medical applications. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1104–1114, 2024.

Zefang Liu and Jiahua Luo. AdaMoLE: Fine-tuning large language models with adaptive mixture of low-rank adaptation experts. In *First Conference on Language Modeling*, 2024.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Ang Lv, Jin Ma, Yiyuan Ma, and Siyuan Qiao. Coupling experts and routers in mixture-of-experts via an auxiliary loss. In *International Conference on Learning Representations*, pp. 75251–75271, 2026a.

Bo Lv, Chen Tang, Zifan Zheng, Bohao Yang, Kun Zhao, Ning Liao, Xiaoxing Wang, Feiyu Xiong, Zhiyu Li, Nayu Liu, and Jingchi Jiang. GRAPHMOE: Amplifying cognitive depth of mixture-of-experts network via introducing self-rethinking mechanism. *arXiv preprint arXiv:2501.07890v4*, 2026b.

Lin Mu, Haiyang Wang, Li Ni, Lei Sang, Zhize Wu, Peiquan Jin, and Yiwen Zhang. TalkLoRA: Communication-aware mixture of low-rank adaptation for large language models. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 18436–18451, 2026.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4885–4901, 2020.

Stefan K. Nielsen, Rachel S. Y. Teo, Laziz U. Abdullaev, and Tan M. Nguyen. Tight clusters make specialized experts. In *International Conference on Learning Representations*, pp. 53891–53917, 2025.

Lin Ning, Harsh Lara, Meiqi Guo, and Abhinav Rastogi. MoDE: Effective multi-task parameter efficient fine-tuning with a mixture of dyadic experts. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 8248–8261, 2025.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 8732–8740, 2020.

Boyan Shi, Wei Chen, Shuyuan Zhao, Junfeng Shen, Shengnan Guo, Shaojiang Wang, and Huaiyu Wan. SAMoRA: Semantic-aware mixture of LoRA experts for task-adaptive learning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 28173–28188, 2026.

Kai Sun, Dian Yu, Jianshu Chen, Dong Yu, Yejin Choi, and Claire Cardie. DREAM: A challenge data set and models for dialogue-based reading comprehension. *Transactions of the Association for Computational Linguistics*, 7:217–231, 2019.

Chunlin Tian, Zhan Shi, Zhijiang Guo, Li Li, and Chengzhong Xu. HydraLoRA: An asymmetric LoRA architecture for efficient fine-tuning. In *Advances in Neural Information Processing Systems*, pp. 9565–9584, 2024.

Zichen Tian, Antoine Ledent, and Qianru Sun. Scalable multi-task low-rank model adaptation. In *International Conference on Learning Representations*, pp. 74390–74410, 2026.

Xun Wu, Shaohan Huang, and Furu Wei. Mixture of LoRA experts. In *International Conference on Learning Representations*, pp. 47302–47318, 2024.

Hengyuan Xu, Wenjun Ke, Yao He, Jiajun Liu, Dong Nie, Peng Wang, Ziyu Shang, and Zijie Xu. Optimizing LoRA allocation of MoE with the alignment of topic correlation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 27251–27259, 2026.

Jingwei Xu, Junyu Lai, and Yunpeng Huang. MeteoRA: Multiple-tasks embedded LoRA for large language models. In *International Conference on Learning Representations*, pp. 55049–55076, 2025.

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. *arXiv preprint arXiv:2407.10671*, 2024.

Ted Zadouri, Ahmet Üstün, Arash Ahmadian, Beyza Ermiş, Acyr Locatelli, and Sara Hooker. Pushing mixture of experts to the limit: Extremely parameter efficient MoE for instruction tuning. In *International Conference on Learning Representations*, pp. 25401–25420, 2024.

Yuan Zhuang, Yi Shen, Yuexin Bian, Qing Su, Shihao Ji, Yuanyuan Shi, and Fei Miao. LD-MoLE: Learnable dynamic routing for mixture of LoRA experts. In *International Conference on Learning Representations*, pp. 64840–64858, 2026.

Jianhui Zuo, Xuemeng Song, Haokun Wen, Meng Liu, Yupeng Hu, Jiuru Wang, and Liqiang Nie. D2MoRA: Diversity-regulated asymmetric MoE-LoRA decomposition for efficient multi-task adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 29286–29294, 2026.

SUPPLEMENTARY MATERIAL

This supplementary material provides additional theoretical analysis, implementation and experimental details, and supplementary experimental results that were omitted from the main paper due to space limitations. These materials further support the theoretical foundation, reproducibility, and empirical findings of COACH. Specifically:

• **Appendix A** provides the theoretical analysis of Prototype-Guided Dimension Reweighting (PDR). Starting from a reverse-KL-regularized prototype-separation objective, we derive its unique optimal dimension-weight solution, establish the connection between this analytical solution and the practical parameterization of PDR used in the main paper, and further analyze the resulting discriminability ordering and prototype-separation properties.

• **Appendix B** provides the implementation and experimental details of COACH, including the statistical updates of expert prototypes in PDR, the construction and normalization of the collaboration prior in CRC, the warm-up and progressive activation schedules, as well as dataset specifications, evaluation protocols, training hyperparameters, and baseline-specific configurations.

• **Appendix C** presents additional experimental results and analyses. In addition to providing more complete experimental protocols and supplementary results for the PDR and CRC mechanistic analyses, the Helper-identity intervention, and the comparison between Anchor-First and Joint-CRC reported in the main paper, this appendix includes several other important experiments that could not be fully presented in the main text due to space constraints. Together, these analyses examine COACH's sensitivity to hyperparameters and the active expert budget, the roles of PDR and CRC, and patterns of routing behavior and expert utilization.

A THEORETICAL ANALYSIS

This appendix provides the proofs of Lemma 1 and Theorem 1 in Section 4, together with the supporting parameterization, dimension-allocation, and metric details. An expert e, a layer l, and a finite snapshot of the running prototypes are fixed throughout. The notation follows Section 4, with expert and layer indices suppressed where unambiguous.

A.1 PROOF OF LEMMA 1

Set $c_q = \sigma_{\max} - \sigma_q \geq 0$. Since $\sum_q p_q = 1$, $\mathbf{c}^\top \mathbf{p} = \sigma_{\max} - G(\mathbf{p})$. Thus, maximizing Eq. (8) is equivalent to minimizing $F_{\mathbf{c}}(\mathbf{p}) = \mathbf{c}^\top \mathbf{p} + \lambda D_{\mathrm{KL}}(\mathbf{u}\|\mathbf{p})$. Expanding the reverse Kullback–Leibler (KL) divergence for the uniform reference $\mathbf{u}$ gives

$$F_{\mathbf{c}}(\mathbf{p}) = \mathbf{c}^\top \mathbf{p} + \lambda D_{\mathrm{KL}}(\mathbf{u}\|\mathbf{p}) = \mathbf{c}^\top \mathbf{p} - \frac{\lambda}{d} \sum_{q=1}^{d} \log p_q - \lambda \log d. \tag{A.1}$$

Its Hessian on the positive orthant is

$$\nabla^2 F_{\mathbf{c}}(\mathbf{p}) = \frac{\lambda}{d} \operatorname{diag}\big(p_1^{-2}, \ldots, p_d^{-2}\big) \succ 0.$$

Hence $F_{\mathbf{c}}$ is strictly convex. Since $\mathbf{c}$ is finite, the linear term is bounded on the closed simplex, whereas the logarithmic term tends to $+\infty$ whenever any coordinate approaches zero. Consequently, the sublevel set

$$\{\mathbf{p} \in \Delta_d^\circ : F_{\mathbf{c}}(\mathbf{p}) \leq F_{\mathbf{c}}(\mathbf{u})\}$$

is nonempty, compact, and contained in the simplex interior. Continuity establishes existence of a minimizer, and strict convexity establishes uniqueness.

Introducing a Lagrange multiplier ν for the constraint $\sum_q p_q = 1$, stationarity gives

$$c_q - \frac{\lambda}{d p_q} + \nu = 0, \qquad p_q = \frac{\lambda/d}{c_q + \nu}. \tag{A.2}$$

Positivity requires $\nu > -\min_q c_q$. Normalization therefore reduces to

$$h(\nu) := \sum_{q=1}^{d} \frac{1}{c_q + \nu} = \frac{d}{\lambda}. \tag{A.3}$$

On this interval, h is continuous and strictly decreasing because

$$h'(\nu) = -\sum_{q=1}^{d} \frac{1}{(c_q + \nu)^2} < 0.$$

Moreover, $h(\nu)$ tends to $+\infty$ as $\nu \downarrow -\min_q c_q$, and to zero as $\nu \to +\infty$. Thus, Eq. (A.3) has exactly one feasible solution. The resulting vector is positive, normalized, and stationary. Setting $\alpha = \sigma_{\max} + \nu$ gives $c_q + \nu = \alpha - \sigma_q$. Since $\min_q c_q = 0$, the unique feasible ν yields a unique $\alpha > \sigma_{\max}$. With $Z = \sum_q (\alpha - \sigma_q)^{-1}$, normalization gives $Z = d/\lambda$, and hence $p_q^\star = 1/[Z(\alpha - \sigma_q)]$. This proves Eq. (9) and the existence and uniqueness asserted in Lemma 1. □

A.2 Prototype Separation and Practical Parameterization

Proof of Theorem 1. The parameterization in Eq. (2) satisfies $\alpha > \sigma_{\max}$ because $\delta_\alpha \geq 0$ and $\epsilon > 0$. Thus, $Z = \sum_q (\alpha - \sigma_q)^{-1}$ is finite and positive, and $\lambda_{\text{eff}} = d/Z > 0$. Applying Lemma 1 with $\lambda = \lambda_{\text{eff}}$ shows that the normalized PDR weights in Eq. (2) uniquely maximize Eq. (8) at this induced regularization strength.

Comparing the optimal objective with the feasible uniform vector $\mathbf{u}$ yields

$$G(\mathbf{p}^\star) - \lambda_{\text{eff}} D_{\text{KL}}(\mathbf{u} \| \mathbf{p}^\star) \geq G(\mathbf{u}). \tag{A.4}$$

By linearity, $G(\bar{\mathbf{p}}) - G(\mathbf{u}) = \rho_l [G(\mathbf{p}^\star) - G(\mathbf{u})]$. Combining this identity with Eq. (A.4) and the nonnegativity of KL divergence proves Eq. (11).

To establish the strictness condition and the exact gain decomposition, first note that

$$p_q^\star - p_r^\star = \frac{\sigma_q - \sigma_r}{Z(\alpha - \sigma_q)(\alpha - \sigma_r)}. \tag{A.5}$$

Expanding the pairwise sum and using $\sum_q p_q^\star = 1$ gives

$$\begin{aligned} \sum_{q<r} (\sigma_q - \sigma_r)(p_q^\star - p_r^\star) &= \frac{1}{2} \sum_{q,r} (\sigma_q - \sigma_r)(p_q^\star - p_r^\star) \\ &= d \sum_q \sigma_q p_q^\star - \left(\sum_q \sigma_q \right) \left(\sum_r p_r^\star \right) \\ &= d \big[G(\mathbf{p}^\star) - G(\mathbf{u}) \big]. \end{aligned} \tag{A.6}$$

Substituting Eq. (A.5) into Eq. (A.6) and applying the interpolation identity yields

$$G(\bar{\mathbf{p}}) - G(\mathbf{u}) = \frac{\rho_l}{dZ} \sum_{q<r} \frac{(\sigma_q - \sigma_r)^2}{(\alpha - \sigma_q)(\alpha - \sigma_r)}. \tag{A.7}$$

Every denominator is positive. Hence the gain is strictly positive if and only if $\rho_l > 0$ and at least two discriminability scores differ. For $d = 1$, the pairwise sum is empty and the gain is zero, consistently with this condition. This completes the proof. □

Practical parameterization. The construction in Eq. (2) satisfies

$$\alpha - \sigma_q = (\sigma_{\max} - \sigma_q) + \delta_\alpha (\sigma_{\max} - \bar{\sigma}) + 2\epsilon \geq 2\epsilon > 0, \tag{A.8}$$

because $\delta_\alpha \geq 0$ and $\epsilon > 0$. Thus, the normalized weights in Eq. (2) coincide with $\mathbf{p}^\star$ in Eq. (9), and Theorem 1 applies with the corresponding λ_{eff}. Rather than solving for α under a prescribed regularization coefficient, the practical construction specifies α directly and thereby determines λ_{eff}. This coefficient depends on the current prototype statistics and may vary across experts, layers, and updates.

For fixed, nonconstant scores, increasing δ_α increases α. For any $\sigma_q > \sigma_r$,

$$\frac{p_q^\star}{p_r^\star} = \frac{\alpha - \sigma_r}{\alpha - \sigma_q} = 1 + \frac{\sigma_q - \sigma_r}{\alpha - \sigma_q}. \tag{A.9}$$

This ratio decreases toward one as α increases. Consequently, larger δ_α produces flatter normalized weights, with $\mathbf{p}^\star \to \mathbf{u}$ as $\delta_\alpha \to \infty$.

Dimension allocation. Equation (A.5) directly establishes the ordering and equality relations in Eq. (12). For the interpolated routing weights $\mathbf{v} = d\bar{\mathbf{p}}$,

$$v_q - v_r = \rho_l d(p_q^\star - p_r^\star).$$

Thus, strict discriminability ordering is retained when $\rho_l > 0$, while equally discriminative dimensions receive equal weights. If all scores coincide, normalization gives $\mathbf{p}^\star = \mathbf{u}$ and hence $\mathbf{v} = \mathbf{1}$. At $\rho_l = 0$, identity weighting is recovered regardless of the scores.

Metric correspondence. Since $\bar{\mathbf{p}}$ is a convex combination of positive simplex vectors, the pre-clipping weights satisfy $v_q > 0$ and $\sum_q v_q = d$. Define the expert-specific diagonal matrix $\mathbf{D}_e = \mathrm{diag}(\mathbf{v})$, which therefore has $\mathrm{tr}(\mathbf{D}_e) = d$. By Eq. (1),

$$\begin{aligned}\sum_{j\neq e} \left\| \mathbf{D}_e^{1/2}(\boldsymbol{\mu}_e - \boldsymbol{\mu}_j) \right\|_2^2 &= \sum_{q=1}^{d} v_q \sigma_q = dG(\bar{\mathbf{p}}) \\ &\geq dG(\mathbf{u}) = \sum_{j\neq e} \|\boldsymbol{\mu}_e - \boldsymbol{\mu}_j\|_2^2.\end{aligned} \tag{A.10}$$

Thus, the comparison with identity weighting holds at the same total metric weight, excluding gains caused solely by uniform rescaling.

For the pre-clipping form of Eq. (3), the bias-free routing term satisfies

$$\mathbf{w}_e^\top \mathbf{D}_e \mathbf{x} = \left(\mathbf{D}_e^{1/2}\mathbf{w}_e\right)^\top \left(\mathbf{D}_e^{1/2}\mathbf{x}\right). \tag{A.11}$$

The same expert-specific diagonal metric therefore appears in both the prototype-separation criterion and the routing score's bilinear form.

Scope of the guarantees. The optimality result characterizes $\mathbf{p}^\star$, while the separation and allocation statements also cover the interpolated weights before clipping. With the activation coefficient $\beta_t \in [0, 1]$ in Appendix B.1, replacing ρ_l by $\beta_t \rho_l$ gives the same pre-clipping results; strict separation gain requires nonconstant scores and $\beta_t \rho_l > 0$. Clipping to common positive bounds preserves positivity and non-strict coordinate ordering, but can change the total weight mass. The fixed-mass separation guarantee therefore does not automatically extend to the clipped weights. This analysis establishes a prototype-space separation property; effects on learned routing margins, Anchor stability, and downstream performance remain empirical questions.

B Implementation and Experimental Details

B.1 Implementation Details of COACH

Prototype Estimation and Update. Each routed Attention module maintains independent expert prototypes, initialized from its normalized router weights. Let $\mathcal{N}(x) = x/\max(\|x\|_2, 10^{-6})$. For a feature h_n after router dropout, we use $\mathcal{N}(h_n)^\top \mathcal{N}(w_e)$ to determine a proxy Top-2 set and apply a softmax to these proxy scores divided by τ_l within that set, obtaining assignments $\pi_{n,e}$ (zero outside the set). Omitting layer superscripts, the update is

$$\begin{aligned} m_e &= \sum_n \pi_{n,e}, \qquad \bar{\mu}_e = \mathcal{N}\left(\frac{\sum_n \pi_{n,e}\mathcal{N}(h_n)}{\max(m_e, 10^{-6})}\right), \\ \mu_e &\leftarrow \mathcal{N}(0.99\mu_e + 0.01\bar{\mu}_e). \end{aligned} \tag{B.1}$$

Experts with $m_e < 10^{-6}$ retain their prototypes. Statistics cover all router positions, including prompt, answer, and padding tokens. Updates begin with the first training forward and are eligible every four optimizer steps; all four accumulated microbatches within an eligible step update the prototypes sequentially. Updates use detached features and assignments and are computed in FP32 under `no_grad`. Prototypes are stored as persistent buffers in the model dtype and reused between updates.

PDR Implementation. Prototype discriminability and reciprocal weights follow Eqs. (1)–(2), with $\delta_\alpha = 1.0$ and $\epsilon = 10^{-6}$. For layer $l \in \{1, \ldots, L\}$, $\rho_l = (l-1)/(L-1)$, where $L = 28$. Given the activation coefficient β_t below, the effective weights are

$$v_{e,q}^{(l)}(t) = \mathrm{clip}\Big(1 + \beta_t \rho_l \big(d\, p_{e,q}^{(l)} - 1\big),\, 0.5,\, 2.0\Big). \tag{B.2}$$

Here $p_{e,q}^{(l)}$ is the dimension-weight distribution in Eq. (2). Denominators are lower-bounded by ϵ; non-finite weights use identity or clipping-bound fallbacks. The weights modify the affine router scores for all token–expert pairs, while LoRA experts receive the original representations. During warm-up or under degenerate discriminability, the weights reduce to one. Cached weights are recomputed when the optimizer step or prototype-update count changes and reused otherwise.

CRC Statistics and Prior Construction. Using the current prototypes and LoRA matrices, CRC computes the response $C_{i,j}$ in Eq. (4). With layer superscripts omitted, the normalization is

$$M_{i,j} = \log \max(C_{i,j}, \epsilon_C), \qquad S_{i,j} = (M_{i,j} - \bar{M}_i)\, \mathbf{1}[j \neq i], \tag{B.3}$$

$$H_{i,j} = \begin{cases} \dfrac{S_{i,j} - \bar{S}_i}{\max(q_i, \epsilon_C)}, & j \neq i,\ q_i > \epsilon_C, \\ 0, & \text{otherwise.} \end{cases} \tag{B.4}$$

Here $\bar{M}_i$ and $\bar{S}_i$ are the off-diagonal row means, q_i is the population standard deviation of the off-diagonal entries of S_i, and $\epsilon_C = 10^{-8}$. All row statistics exclude the diagonal; $H_{i,i}$ and degenerate rows are explicitly set to zero, and non-finite intermediate values are sanitized. Each routed module constructs its prior in FP32 under `no_grad`, on first use and every 20 optimizer steps thereafter. Cached priors are reused between refreshes. Refreshing precedes the current forward's prototype update; fresh checkpoint evaluation rebuilds the prior. Without a valid prior, selection falls back to Top-2 routing without CRC calibration.

Warm-up and Progressive Activation. Let t count completed optimizer updates. Prototype and CRC statistics accumulate from the start of training. PDR and CRC share a 100-step warm-up followed by a 100-step linear ramp:

$$\beta_t = \mathrm{clip}\left(\frac{t - 100}{100}, 0, 1\right), \qquad \gamma_t = 0.25\, \beta_t. \tag{B.5}$$

Both calibrations are inactive for $t < 100$ and reach full strength at $t \geq 200$. We apply no subsequent annealing or freezing of these statistics.

Routing Temperature and Final Sparse Aggregation. We retain the Anchor and select $K - 1$ Helpers from the remaining experts, with $K = 2$ in the main experiments. The final weights are normalized only over the active set:

$$g_{n,e}^{(l)} = \frac{\exp(s_{n,e}^{(l)}/\tau_l)}{\sum_{j \in \mathcal{S}_n^{(l)}} \exp(s_{n,j}^{(l)}/\tau_l)}, \qquad e \in \mathcal{S}_n^{(l)}, \tag{B.6}$$
$$\tau_l = \max(1.5 - 0.8\rho_l, 10^{-8}).$$

Weights outside the set are zero. The implementation uses a numerically stable softmax, with the layer temperature active throughout training, including calibration warm-up.

B.2 Experimental Setup

Benchmarking Datasets. Evaluation covers seven tasks: QASC, DREAM, CosmosQA, MuTual, WinoGrande, ART, and ANLI, matching Section 5.1 and Tables 1 and 2. Table 3 gives their sizes after preprocessing. The final evaluation protocol uses validation/development data for checkpoint selection and independent official test splits for performance assessment. DREAM and MuTual use development splits for validation; ANLI combines R1–R3 within each split, and WinoGrande uses `winogrande_xl`. Preprocessing removes identical MuTual mirror duplicates and one ANLI training example overlapping with validation. Test data are excluded from training, checkpoint selection, and hyperparameter tuning.

Table 3: Sizes of the seven benchmark datasets after preprocessing. Validation/development data support checkpoint selection; independent test data support final performance evaluation. Totals cover the listed datasets.

Dataset	QASC	DREAM	CosmosQA	MuTual	WinoGrande	ART	ANLI	Total
Train	8,134	6,116	25,262	7,088	40,398	169,654	162,864	**419,516**
Validation	926	2,040	2,985	886	1,267	1,532	3,200	**12,836**
Test	920	2,041	6,963	886	1,767	3,040	3,200	**18,817**

Evaluation Protocol and Metrics. The Qwen chat template presents the instruction, question/context, and lettered choices, requesting one answer letter. Training supervises only the gold letter and end-of-text token. Evaluation selects the valid candidate letter with the highest next-token logit; all labels are single tokens, so no length normalization is used. For final task accuracy Acc_t over N_t independent test examples in each of the seven reported tasks,

$$\text{Macro} = \frac{1}{7}\sum_{t=1}^{7}\mathrm{Acc}_t, \qquad \text{Micro} = \frac{\sum_{t=1}^{7} N_t \mathrm{Acc}_t}{\sum_{t=1}^{7} N_t}, \qquad \sum_{t=1}^{7} N_t = 18{,}817. \tag{B.7}$$

The seven-task test metrics are computed separately for each seed and then averaged over multiple shared random seeds, with sample standard deviations across runs. Validation occurs every 1,000 steps. The existing validation-best checkpoint from each run is retained, reloaded, and evaluated on the independent test splits without further training or checkpoint selection on test data. DREAM and ANLI provide public test labels; QASC, CosmosQA, MuTual, WinoGrande, and ART require official scoring of test predictions.

Training Configuration. COACH uses Qwen2-1.5B Base and the adapter configuration in Section 5.1, with LoRA scaling $s_{\mathrm{L}} = \alpha/\sqrt{r}$. Only routers and LoRA parameters are trained. Each microbatch samples a task uniformly, then draws examples with replacement from its full training split; no per-task sample caps are applied. Training uses AdamW for 10,000 optimizer updates, weight decay 0.01, effective batch size 32 (microbatch 8, accumulation 4), maximum training length 1,024 tokens including source and target, and BF16 precision. Adapter/router dropout and label smoothing are 0.1. All learning rates warm linearly to 2×10^{-5} over 100 steps and remain there through step 500; thereafter, Attention experts and FFN LoRA use 10^{-4}, while routers retain 2×10^{-5}. Routing and statistical-update settings are given in Section B.1.

Implementation of Compared Methods. All methods share the backbone, data, evaluation protocol, and 10,000 optimizer updates. LoRA uses Microsoft's released implementation; other methods adapt released implementations to the shared framework while retaining their module placement and expert parameterization. Table 4 summarizes key differences. Configurations are fixed across seeds; differing effective batch sizes mean the budget is matched in optimizer updates, not sampled examples.

HMoRA uses shared A matrices and QKV/FFN routers with divergence regularization. HotMoE uses EE/TE routing with an auxiliary objective; mtLoRA uses load balancing and spectral regularization. The remaining baselines have no auxiliary objectives. TalkLoRA applies no additional forward-pass LoRA scaling.

C Supplementary Experimental Results

The main and component-ablation comparisons in Tables 1 and 2 use independent test splits of seven benchmarks, as specified in Appendix B.2. The hyperparameter and mechanistic analyses, together with the additional Top-3 and Top-4 runs, use the seven-task validation set of 12,836 examples. Their aggregate accuracies use validation-set sizes, and their token-level diagnostics use the evaluation inputs specified below. The Top-2 block of Table 6 reproduces the independent-test results from Table 1.

Table 4: Key baseline configurations. b is effective batch size; linear and constant denote learning-rate schedules after warm-up.

Method	Adapters / routing	Optimization
LoRA	q/v; $r = 8$, $\alpha = 16$.	LR 2×10^{-4}; linear; $b = 16$.
MoLoRA	$q/k/v/o$, gate/up/down; 8 dense experts; $r = 8$, $\alpha = 16$; scaling α/r.	COACH's two-stage LR schedule; dropout 0; $b = 32$.
TalkLoRA	$q/k/v$, up/down; 4 dense experts; total rank 32; cross-layer shared B-banks.	LR 3×10^{-4}; linear; $b = 16$.
HMoRA	MoE on $q/k/v$, gate/up; single LoRA on o/down; 8 experts, Top-2; $r = 8$, $\alpha = 8$.	Base LR 5×10^{-5}, B matrices 6×10^{-5}; constant; $b = 16$.
HotMoE	gate/up/down; 5 experts, Top-2; $r = 8$, $\alpha = 16$.	LR 5×10^{-5}; constant; $b = 32$.
mtLoRA	Parallel FFN-block adapter; 16 experts, dense group routing; $r = 16$, $\alpha = 64$.	LR 2×10^{-4}; linear, no warm-up; $b = 32$.

C.1 Detailed Main and Ablation Results

This section provides additional discussion of the main and component ablation results in Tables 1 and 2, and complete numerical results for the hyperparameter sensitivity analysis in Figure 6. Main and component experiments use multiple shared random seeds. Under the final evaluation protocol, each run's validation-best checkpoint is reloaded and scored on the independent test splits. Seven-task scores are averaged across runs, and sample standard deviations use the $R - 1$ denominator, where R is the number of runs. The hyperparameter analysis retains a single random seed and reports aggregate validation accuracy across the seven tasks. Training and candidate-answer scoring follow Appendix B.2, with these distinct evaluation splits. All accuracies are percentages.

Detailed Main Results. Table 1 reports task-wise and aggregate means, with sample standard deviations for task-wise, Macro, and Micro Accuracy. Methods share the training data, evaluation protocol, and budget of 10,000 optimizer updates while retaining their method-specific configurations, including effective batch sizes, as detailed in Appendix B.2.

Averaged over the shared runs, COACH achieves the highest Macro and Micro Accuracy, reaching $76.98 \pm 0.16\%$ and $76.79 \pm 0.22\%$, respectively. The mean improvements over mtLoRA, the strongest baseline in aggregate performance, are 0.52 and 0.49 percentage points. COACH ranks first on six of the seven reported tasks and second on ANLI. The standard deviations describe variation across training runs.

Detailed Component Ablations. Table 2 reports task-wise and aggregate means, with sample standard deviations for task-wise, Macro, and Micro Accuracy. The configuration with both PDR and CRC disabled is denoted as **Base** and compared with PDR-only, CRC-only, and the full COACH model. All four configurations use the same number of experts, LoRA rank, and Top-2 activation budget, differing only in whether PDR and CRC are enabled. PDR-only applies prototype-guided dimension reweighting followed by standard Top-2 selection. CRC-only determines the Anchor from routing scores computed without PDR and subsequently selects the Helper conditioned on that Anchor.

Here, Base is a structurally matched ablation baseline used to isolate component contributions, rather than the MoLoRA method (Zadouri et al., 2024) evaluated in Table 1. MoLoRA is an independent prior method with a different configuration and a larger parameter count in these experiments; the two therefore serve different comparison purposes.

Across the shared runs, enabling either PDR or CRC improves mean Macro Accuracy over Base by 1.83 and 1.74 percentage points, respectively. Combining both components yields $76.98 \pm 0.16\%$ Macro Accuracy and $76.79 \pm 0.22\%$ Micro Accuracy, improving the means over Base by 3.10 and 3.15 percentage points and outperforming either component alone. These results support the complementary contributions of PDR and CRC to overall performance. Their specific effects on routing behavior are examined in Appendix C.3 and C.4, respectively.

Table 5: **Hyperparameter sensitivity of COACH on seven tasks.** Results report seven-task Macro Accuracy from a single random seed. Each sweep varies one hyperparameter; its best Macro Accuracy is bolded.

Hyperparameter sweep	γ	δ_α	Macro Accuracy (%)
CRC strength γ	0.0625	1.0	76.53
	0.125	1.0	76.41
	0.25	1.0	**77.03**
	0.5	1.0	76.58
	1.0	1.0	76.62
	1.5	1.0	76.87
	2.0	1.0	76.75
	4.0	1.0	76.75
PDR parameter δ_α	0.25	0.125	76.95
	0.25	0.25	76.85
	0.25	0.5	76.62
	0.25	**1.0**	**77.03**
	0.25	2.0	76.70
	0.25	4.0	76.96
	0.25	8.0	76.84

Table 6: **Seven-task performance under different active expert budgets.** K denotes COACH's active-expert budget ($E = 8$); baselines use their method-specific configurations. The $K = 2$ block reports mean independent-test accuracies over multiple shared random seeds. The $K = 3$ and $K = 4$ blocks report validation accuracies at the validation-best checkpoints from single runs using the same shared random seed. All accuracies are in %.

K	Metric	LoRA	MoLoRA	TalkLoRA	HMoRA	mtLoRA	HotMoE	COACH
2	Macro	72.75	76.37	74.80	74.54	76.47	75.75	**76.98**
	Micro	71.85	76.18	73.90	74.02	76.30	75.86	**76.79**
3	Macro	73.30	76.63	75.47	74.83	76.98	76.12	**77.17**
	Micro	70.80	74.63	73.26	72.80	74.86	73.67	**75.02**
4	Macro	73.08	76.27	75.70	74.77	76.51	75.86	**76.82**
	Micro	70.36	73.82	73.28	72.47	74.14	73.86	**74.49**

Complete Hyperparameter Sensitivity Results. Table 5 provides the numerical results corresponding to Figure 6. The analysis examines the CRC collaboration strength γ and the PDR dimension-reweighting parameter δ_α. When γ varies, $\delta_\alpha = 1.0$ is fixed; when δ_α varies, $\gamma = 0.25$ is fixed. All other training settings remain unchanged. Both sweeps use the same random seed and reuse its default COACH result at $(\gamma, \delta_\alpha) = (0.25, 1.0)$.

Within the tested ranges, the default configuration, $\gamma = 0.25$ and $\delta_\alpha = 1.0$, achieves the highest Macro Accuracy in both sweeps, supporting the choice of moderate calibration strengths for this random seed. Table 5 reports single-seed validation accuracy for hyperparameter selection, whereas Table 1 reports accuracy on independent test splits averaged across multiple shared random seeds.

C.2 Effect of the Active Expert Budget

COACH uses Top-2 routing in the main experiments. To examine whether the Anchor–Helper design remains effective when several Helpers are activated jointly, Top-3 and Top-4 settings are additionally evaluated. These experiments use the same Qwen2-1.5B backbone and the same seven-task evaluation scope as the main results, with $E = 8$ experts and LoRA rank $r = 8$. COACH retains the default hyperparameters $\delta_\alpha = 1.0$ and $\gamma = 0.25$. Under Top-3 and Top-4 routing, it selects one Anchor

followed by two and three Helpers, respectively, using the same Anchor-conditioned candidate scores and normalizes the final routing weights only over the active experts.

Table 6 combines mean independent-test accuracies over multiple shared random seeds at Top-2 with single-run validation results at Top-3 and Top-4. All three blocks cover the same seven tasks and compare LoRA, MoLoRA, TalkLoRA, HMoRA, mtLoRA, HotMoE, and COACH; the different splits and seed aggregation should be considered when comparing blocks.

At $K = 2$, COACH achieves 76.98% Macro Accuracy and 76.79% Micro Accuracy, outperforming the strongest baseline, mtLoRA, by 0.52 and 0.49 percentage points, respectively. At $K = 3$, COACH reaches 77.17% Macro Accuracy and 75.02% Micro Accuracy, achieving the highest values on both metrics and exceeding mtLoRA by 0.19 and 0.16 percentage points, respectively. At $K = 4$, COACH reaches 76.82% Macro Accuracy and 74.49% Micro Accuracy, obtaining the highest values among the recorded methods and improving over mtLoRA by 0.31 and 0.35 percentage points, respectively. On the validation set with the same random seed as the Top-3 and Top-4 runs, COACH obtains 77.03% Macro and 74.69% Micro Accuracy at $K = 2$. These validation runs indicate that Anchor-conditioned routing remains effective with two or three jointly selected Helpers, while performance does not improve monotonically with the active-expert budget.

C.3 Extended Analysis of PDR

This section supplements Section 5.3.1 and Figure 3 with experimental definitions and complete numerical results, examining the relationships among prototype discriminability, routing margins, and Anchor stability under perturbations.

Analysis setup. The analysis uses the validation-best checkpoint of Qwen2-1.5B COACH trained with a single random seed. Valid token representations are collected from the seven-task validation set of 12,836 examples across all 112 routing modules. For the routing-margin and perturbation experiments, Standard and PDR use the same frozen checkpoint and identical hidden representations before dimension reweighting. Here, Standard denotes routing scores computed without dimension reweighting, rather than the independently trained Base model in Appendix C.1. All analyses concern the base routing scores and Anchor decisions before CRC calibration. All 95% confidence intervals are estimated using 10,000 block-bootstrap resamples, with the resampling unit specified for each analysis.

Prototype discriminability and held-out expert separability. Held-out tokens are grouped by their PDR Anchor assignments, and expert separability is measured along each hidden dimension. Omitting the routing-module index, let $\mathcal{A}$ denote the set of experts with nonempty held-out Anchor groups, and let $M = |\mathcal{A}|$. For $e \in \mathcal{A}$, the separability metric is defined as

$$F_{e,q} = \frac{\dfrac{1}{M-1} \displaystyle\sum_{\substack{j \in \mathcal{A} \\ j \neq e}} (\bar{x}_{e,q} - \bar{x}_{j,q})^2}{\dfrac{1}{M} \displaystyle\sum_{j \in \mathcal{A}} \widehat{\mathrm{Var}}(x_q \mid a = j) + \epsilon}. \tag{C.1}$$

Here, $\bar{x}_{e,q}$ is the mean representation of expert group e along dimension q, within-group variances use the population-variance convention, and $\epsilon = 10^{-12}$. This metric measures between-group centroid separation relative to within-group dispersion. Within each Router–Expert block, it is normalized as $\widetilde{F}_{e,q} = F_{e,q} / (\frac{1}{d} \sum_{s=1}^{d} F_{e,s} + \epsilon)$, and dimensions are partitioned into five approximately equal-sized rank quantiles according to the prototype discriminability score $\sigma_{e,q}$ in Eq. (1).

Of the 896 expected Router–Expert blocks, two contain no held-out Anchor tokens. These empty groups are not imputed; the following supplementary association analysis is conditional on the observed support and uses the remaining 894 nonempty blocks.

As shown in Table 7, separability increases monotonically across discriminability quantiles, with $Q_5 - Q_1 = 1.0963$ and a 95% confidence interval of $[1.0633, 1.1292]$, obtained by resampling Router–Expert blocks. The mean within-block Spearman correlation is 0.3716, with positive correlations in 893 of 894 blocks. These results support a consistent positive association within the observed Anchor support, but do not establish the causal importance of individual dimensions.

Table 7: **Held-out expert separability across prototype-discriminability quantiles.** Quantiles are constructed independently within each Router–Expert block. Values are obtained by averaging normalized separability within each quantile and then equally averaging across blocks.

Discriminability quantile	**Q1 (lowest)**	**Q2**	**Q3**	**Q4**	**Q5 (highest)**
Mean normalized separability	0.5819	0.7595	0.9081	1.0737	1.6782

Routing margins under matched Anchors. Routing margins are compared only on events where Standard and PDR select the same Anchor on the clean input. For scoring method $c \in \{\mathrm{Std}, \mathrm{PDR}\}$, the routing margin is defined as

$$m_c(h) = \frac{z^c_{e^c_1}(h) - z^c_{e^c_2}(h)}{\max\left(\left\| d^c_{e^c_1} - d^c_{e^c_2} \right\|_2, 10^{-12}\right)}. \tag{C.2}$$

Here, e^c_1 and e^c_2 are the highest- and second-highest-scoring experts under method c. The effective routing directions are $d^{\mathrm{Std}}_e = W_e$ and $d^{\mathrm{PDR}}_e = W_e \odot v^{\mathrm{eff}}_e$. Thus, the comparison fixes the Anchor identity while allowing the second-ranked expert to differ between the two methods.

Table 8: **Mean normalized routing-margin gains under matched clean Anchors.** Gains are computed as PDR minus Standard and averaged over eligible routing events within each scope. Layer indices are zero-based.

Scope	**Overall**	**Shallow (0–6)**	**Mid-shallow (7–13)**	**Mid-deep (14–20)**	**Deep (21–27)**
Mean routing-margin gain	$+0.0224$	$+0.0036$	$+0.0110$	$+0.0225$	$+0.0556$

The two scoring methods retain the same Anchor in 92.77% of routing events. On these events, the mean normalized margin increases from 0.3333 to 0.3557, yielding a gain of 0.0224 with a router-bootstrap 95% confidence interval of $[0.0169, 0.0293]$. All four depth groups in Table 8 exhibit positive gains, which increase with depth, consistent with the depth-dependent strengthening of PDR reweighting.

Anchor stability under perturbations. For events with matched clean Anchors, RMS-scaled Gaussian perturbations are applied to the hidden representations:

$$\widetilde{h} = h + \eta\ \mathrm{RMS}(h)\,\xi, \qquad \xi \sim \mathcal{N}(0, I), \tag{C.3}$$

where η controls the perturbation strength. Standard and PDR receive identical noise, and the same base noise is scaled across perturbation strengths. An Anchor flip occurs when a competing expert scores higher than the shared clean Anchor after perturbation.

For each router, 4,096 eligible events are deterministically sampled, and four independent perturbations are drawn per event. Flip rates are computed within each router and then averaged equally across routers. Confidence intervals are obtained by resampling routers.

As shown in Table 9, PDR achieves lower Anchor flip rates at all five perturbation strengths, with an arithmetic mean relative reduction of approximately 6.3%. All 95% confidence intervals for the absolute reductions exclude zero. These results indicate that, under matched clean Anchors and paired Gaussian perturbations to hidden representations, larger routing margins with PDR are accompanied by more stable Anchor decisions.

C.4 Expert Composition and CRC Analysis

This section supplements the experimental protocols for Figures 1(a), 4, and 5. These analyses use the seven-task data scope and use the tokenization and answer-scoring protocol in Appendix B.2; this section specifies the sampling scope, controlled variables, statistical procedures, and comparison policies. The post-hoc diagnostics freeze model parameters and running routing statistics during evaluation. The Anchor-First versus Joint-CRC experiment compares models trained separately under their respective routing policies.

Table 9: **Anchor flip rates under paired Gaussian perturbations.** Absolute reduction is Standard minus PDR, reported in percentage points (pp). Relative reduction uses the Standard flip rate as the denominator. Differences and relative reductions are calculated from unrounded values.

Perturbation strength	Standard flip rate (%)	PDR flip rate (%)	Absolute reduction (pp)	Relative reduction (%)	95% CI for absolute reduction (pp)
2.5%	1.107	1.031	0.076	6.86	[0.044, 0.109]
5%	2.532	2.357	0.175	6.90	[0.134, 0.217]
10%	5.713	5.354	0.358	6.27	[0.294, 0.424]
15%	9.095	8.551	0.544	5.98	[0.459, 0.632]
20%	12.514	11.799	0.715	5.71	[0.610, 0.821]

Composition dependence of expert utility (Figure 1(a)). Local counterfactual analyses are conducted on a frozen Vanilla LoRA-MoE model. The analyses sample 32 validation examples from each of the seven tasks (224 examples in total) and examine the final non-padding token at the q/k/v/o routing modules in layers 7, 14, 21, and 28, yielding 3,584 routing events. For the original Top-2 experts e_1, e_2, each expert is used as the Anchor in turn, and utilities are compared over the same six candidate Helpers, excluding both e_1 and e_2. Both Anchor conditions retain the original Anchor-slot and Helper-slot weights without recomputing softmax. Each counterfactual forward pass modifies only one routing event; all other positions replay the original expert identities and routing weights.

Candidate utility is the negative of the correct-option negative log-likelihood (NLL), with probabilities obtained by normalizing the logits of valid option-letter tokens. The best candidate is the expert with the lowest NLL, with exact ties broken by expert index. The best-candidate change rate uses all routing events as its denominator. For preference reversals, a candidate pair is considered comparable only when its absolute NLL difference exceeds 10^{-3} under both Anchor conditions. The preference reversal rate is the proportion of comparable pairs whose relative ordering reverses, pooling comparable pairs across all events.

Anchor-conditioned Helper distributions (Figure 4(a)). The empirical pairing distributions of independently trained Vanilla LoRA-MoE and full COACH models are compared. Vanilla does not include PDR or CRC. Both models use the same 224 task-balanced validation examples. For each model, ordered Anchor–Helper pairs are counted over all non-padding input tokens to compute

$$Q_{ij} = \frac{N(\text{Anchor} = i, \text{Helper} = j)}{\sum_{k \neq i} N(\text{Anchor} = i, \text{Helper} = k)}, \qquad Q_{ii} = 0. \tag{C.4}$$

This matrix represents the empirical conditional distribution of actual expert selections, not the collaboration prior H used for score calibration.

The figure shows a representative routing module rather than an aggregate over all modules. Among shared modules with observations for all eight Anchors in both models, the selected module is the one whose distribution difference, $\|Q_{\text{COACH}} - Q_{\text{Vanilla}}\|_F$, is closest to the median across these modules. The selected module is the q projection in layer 17, with 37,904 valid token-routing events per model. Because the models differ in training configuration, this visualization describes differences in routing structure but does not isolate the contribution of CRC.

Oracle Helpers and conditional-prior controls (Figure 4(b), left). The analysis uses a frozen full COACH checkpoint to evaluate 3,584 local routing events with the task-balanced sampling size and routing positions specified above. At each event, the Anchor and both slot weights from native COACH routing are fixed, and all seven non-Anchor Helpers are enumerated. All other positions replay the native routing records. The Oracle Helper is the candidate that minimizes the correct-option NLL, using the NLL definition and exact-tie handling described above. The Oracle match rate measures the fraction of events in which a policy selects this exact Oracle identity.

Raw retains the PDR routing scores z but omits CRC calibration. CRC uses the prior row corresponding to the selected Anchor. Global removes Anchor conditioning by constructing a shared Helper

preference from the off-diagonal column means of the frozen prior matrix:

$$b_j = \frac{1}{E-1}\sum_{i\neq j} H_{ij}, \qquad \widetilde{b}_j = \frac{b_j - \mathrm{mean}(b)}{\max\{\mathrm{std}(b), 10^{-8}\}}, \tag{C.5}$$

where the mean and population standard deviation are computed across experts. Global selects the Helper using $z_j + \gamma\widetilde{b}_j$. Shuffled applies a derangement to the Anchor rows, replacing $H_{a,j}$ with $H_{\pi(a),j}$, while leaving Helper columns unchanged and continuing to exclude the actual Anchor a. All policies that use a prior share the same coefficient, $\gamma = 0.25$.

Shuffled uses ten distinct row derangements, each held fixed across all modules and events. The figure reports the mean and population standard deviation across these derangements. Confidence intervals for policy differences use 10,000 paired, example-clustered bootstrap replicates, retaining all routing events from each resampled example. The Oracle uses ground-truth answer information and serves only as a post-hoc diagnostic reference.

Helper-identity intervention (Figure 4(b), right). Two full COACH checkpoints obtained from training runs with independent random seeds are evaluated: one validation-best checkpoint and one final training checkpoint. Each checkpoint is evaluated on the seven-task validation set of 12,836 examples, covering input-token positions across all 112 attention routing modules.

For each batch, the evaluation first runs Raw Top-2 routing, retaining PDR but omitting CRC calibration. At every position, the Anchor, Helper, original slot weights, and the CRC-selected Helper computed from the routing scores on this Raw trajectory are recorded. Intervention forward passes replay the same records, fixing the Anchor and both slot weights at every position while replacing only the Helper identity. The replacement expert inherits the original Helper weight without recomputing softmax. Expert outputs and downstream hidden states change normally, but these changes do not trigger new Anchor selections or routing-weight updates.

Random selects a Helper from all seven non-Anchor experts, allowing the original Helper to be selected again. Five random-replacement evaluations are performed per checkpoint and averaged within each checkpoint, after which the paired changes are averaged equally across the two checkpoints. Raw here is not the independently trained Base in Appendix C.1. This identity intervention also differs from full CRC inference, which recomputes the final mixture weights.

Anchor-First versus Joint-CRC (Figure 5). Anchor-First CRC and Joint-CRC are trained as separate models, with each model's routing policy applied from the beginning of training. Their routers and LoRA experts are optimized under their respective selection rules, and each model maintains its own expert prototypes and collaboration prior H. Each trained model is evaluated using the routing policy employed during its training. At every position, the routing logits z are computed from that model's current hidden representations, and Helper or pair selection uses that model's own collaboration prior.

For a given token and routing module, Joint-CRC searches over all 56 ordered pairs of distinct experts and selects

$$(i^*, j^*) = \arg\max_{i\neq j} \left(z_i + z_j + \gamma H_{i,j}\right). \tag{C.6}$$

Exact ties are resolved using a fixed expert-index ordering. After the pair is selected, its routing weights are computed using the same rule as Anchor-First:

$$(g_{i^*}, g_{j^*}) = \mathrm{softmax}\left(\frac{[\, z_{i^*},\ z_{j^*} + \gamma H_{i^*,j^*}\,]}{\tau_l}\right). \tag{C.7}$$

Both variants use the same active-set normalization, while their learned parameters, routing logits, collaboration priors, and resulting mixture weights are specific to their separately trained models.

Within the Joint-CRC model, let $a = \arg\max_e z_e$ denote the token-specific Top-1 expert before joint pair selection. The fractions of events satisfying $i^* \neq a$ and $a \notin \{i^*, j^*\}$ are measured separately. These statistics distinguish assigning the first position in the ordered pair to a different expert from completely excluding the model's own base Top-1 expert from the active set. Both use all non-padding token–module routing events over the seven-task validation set as their denominator.

C.5 Routing Behavior and Expert Utilization

This section examines whether trained models exhibit unused experts, concentrated expert combinations, or nearly single-expert gating during normal inference, complementing the mechanistic analyses in the main text.

Experimental setup. Pure and COACH are compared on Qwen2-1.5B with eight experts, Top-2 routing, LoRA rank 8, and $\alpha = 16$. Pure is an independently trained reference model that uses standard Top-2 routing without PDR or CRC. The comparison uses one validation-best checkpoint per method, trained with the same random seed. The checkpoints differ in routing temperature and PDR/CRC settings. The comparison therefore describes their native routing behavior rather than isolating the effect of a single component.

Both models receive identical tokenized inputs from the complete seven-task validation set of 12,836 examples. Zero-shot prompts retain the question, context, and candidate options, without appending a correct-answer label or candidate-answer continuation. Each prompt undergoes one forward pass per model without generation. The analysis retains the untruncated inputs used by the executed logit-based checkpoint-selection evaluator: although the run configurations specify a 1,024-token input limit, that evaluator does not apply truncation. Eight DREAM prompts exceed this configured limit, with a maximum length of 1,111 tokens.

The final active expert identities and the mixture weights actually used for output aggregation are collected from all 112 Attention routing modules ($q_{\text{proj}}, k_{\text{proj}}, v_{\text{proj}}, o_{\text{proj}}$ across 28 layers). Model parameters and running routing statistics remain frozen, and no routing intervention is applied. All statistics in this section use all 2,074,399 non-padding prompt tokens, giving 232,332,688 token–module routing events per model.

Diagnostic metrics and aggregation. For each routing module, let N be the total number of non-padding prompt tokens in the validation set and c_e the number of times expert e is selected. The activation-slot share and normalized load entropy are

$$f_e = \frac{c_e}{2N}, \qquad H_{\text{load}} = -\frac{\sum_{e=1}^{E} f_e \log f_e}{\log E}, \tag{C.8}$$

with $0 \log 0 = 0$. This is the entropy of expert selection frequencies, not per-token gating weights. With eight experts and two distinct experts selected per event, $H_{\text{load}} \in [1/3, 1]$; the lower bound is attained when the same expert pair is always selected.

The analysis also reports the number of unused experts $U_0 = \sum_e \mathbf{1}[c_e = 0]$, the maximum unordered expert-pair share $P_{\max}$, and the gate saturation rate R_{sat}. Here, $P_{\max}$ is the fraction of events assigned to the most frequent of the 28 unordered expert pairs, irrespective of Anchor–Helper ordering. R_{sat} is the fraction of events in which the smaller of the two actual normalized mixture weights is below 0.01. Metrics are computed independently within each module and then averaged equally across the 112 modules. Expert identities are never pooled across modules. Within a module, statistics are token-based, so longer inputs contribute more events.

Results on all prompt tokens. Table 10 reports routing diagnostics on all non-padding prompt tokens. COACH has lower mean load entropy than Pure (0.8900 versus 0.9382, a relative decrease of 5.13%) and a higher mean maximum pair share (19.4954% versus 14.1658%, an increase of 5.3296 percentage points). The same directions hold in 93/112 and 92/112 corresponding modules, respectively. Thus, COACH exhibits more concentrated expert selection and pairing in this evaluation, rather than more uniform usage.

Figure 7 shows that concentration is most pronounced in the o_{proj} modules, especially in several later layers. Their mean load entropy is 0.8884 for Pure and 0.8074 for COACH, and their mean maximum pair shares are 20.1286% and 28.8471%, respectively. The largest module-level pair shares are 54.0126% in Pure (layer 21, o_{proj}) and 64.6584% in COACH (layer 23, o_{proj}). COACH has one unused expert in one module (layer 27, o_{proj}), whose entropy is 0.5169; Pure has no zero-use experts on these inputs. All layer indices in this section are zero-based.

Gate saturation remains rare despite the more concentrated selections. Pure has no saturated events, while COACH has 6,233 out of 232,332,688 events (0.002683%). Of these, 5,414 (86.9%) occur

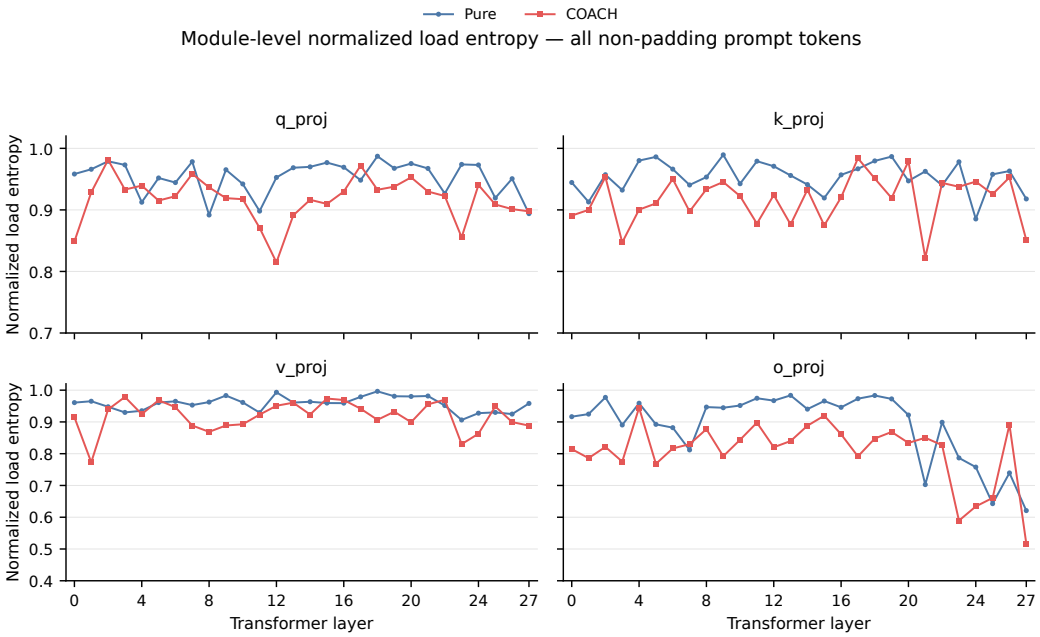


Figure 7: **Module-level normalized load entropy on all non-padding prompt tokens.** Each panel shows one projection type across the 28 Transformer layers for Pure and COACH. Values are computed independently for each routing module on the same validation inputs. Layer indices are zero-based. The curves describe module-level variation.

Table 10: **Routing diagnostics of Pure and COACH on all non-padding prompt tokens.** Each value is an equal-weight mean over 112 routing modules from one checkpoint per method, using a shared training random seed. U_0 is the mean number of unused experts per module; $P_{\max}$ and R_{sat} are percentages. No cross-seed standard deviations are reported.

Method	**Normalized load entropy** H_{load}	**Unused experts per module** U_0	**Maximum pair share** $P_{\max}$ (%)	**Gate saturation rate** R_{sat} (%)
Pure	0.9382	0.0000	14.1658	0.000000
COACH	0.8900	0.0089	19.4954	0.002683

in layer 27's q_{proj}, which has the highest module-level saturation rate, 0.2610%. Thus, localized selection concentration is present, but nearly single-expert gating is uncommon under the measured threshold. The absence of zero-use experts in Pure should not be interpreted as uniformly substantial use: two experts in its layer-27 o_{proj} are selected on only approximately 0.0135% and 0.0172% of valid token events.

Interpretation. Overall, these results indicate that COACH induces a more structured and selective routing pattern than the Pure baseline. The lower load entropy and higher dominant-pair share show that expert usage becomes more concentrated around recurring expert combinations, particularly in deeper output-projection modules. Importantly, this concentration does not arise from effectively suppressing the second selected expert: gate saturation remains negligible, occurring in only 0.0027% of routing events. Together, these observations suggest that COACH promotes structured expert pairing and specialization while preserving genuine multi-expert activation. The increased routing selectivity is therefore expressed primarily through more concentrated expert-pair structure rather than through a collapse toward effectively single-expert gating.