RETHINKING STATE-CONTROL DYNAMICS FOR UNIFIED AND EFFICIENT TRANSFORMER ADAPTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

State-based fine-tuning reduces training memory by injecting lightweight controls into model states rather than modifying pretrained weights. However, these controls generally cannot be merged into the pretrained weights and must be evaluated at every discretization step, limiting inference throughput and practical deployment. Existing approaches incur two avoidable overheads: (1) they employ fine-grained subblock-level discretization, which requires frequent control evaluation, and (2) they instantiate separate controls across layers, forcing multiple control features to share a single routing during communication, limiting adaptation flexibility. In this work, we introduce **SURF** (**S**imilarity-Guided **U**nified **R**outing of **F**eatures), a unified state-based adaptation framework that integrates coarse-grained discretization with similarity-guided routing of shared control features. SURF composes consecutive residual subblocks into layer-wise transitions, reducing control evaluations without altering the pretrained transformations. It then groups layers with similar representations and applies sparse, token-dependent routing over a shared pool of rank-one control features, enabling feature reuse across depth while preserving input-specific adaptation. Experiments on 34 datasets and 10 pretrained models show that SURF substantially narrows the inference-efficiency gap with mergeable weight-based methods while matching or outperforming existing state-based approaches at comparable training cost. These results highlight discretization granularity and control sharing as two key design principles for efficient state-based adaptation.

1 INTRODUCTION

Fine-tuning foundation models is effective but costly because each task requires updating and storing a full model. Parameter-efficient fine-tuning (PEFT) reduces this burden by freezing the backbone and learning lightweight task-specific parameters. LoRA (Hu et al., 2022) uses low-rank weight updates, while variants such as VeRA (Kopiczko et al., 2023), BinaryLoRA (Zhang et al., 2024a), MoLE (Wu et al., 2024), DoRA (Liu et al., 2024), and MoDE (Ning et al., 2025) further improve the performance–efficiency trade-off. However, reducing the number of trainable parameters does not necessarily translate into comparable savings in activation memory or runtime. Because weight-based PEFT still retains intermediate activations for storage and backpropagation, lowering adapter rank may yield only modest memory savings; for example, halving DoRA's rank saves only 0.256 GB on an 8B model (Zhang et al., 2025). In addition, most methods use layer-specific adapters, limiting information sharing across depth. MoLEx (Teo & Nguyen, 2025) enables cross-layer interaction, but requires at least twice as many layer forward passes, leading to substantially higher computational cost as model depth and layer complexity increase.

These limitations motivate state-based fine-tuning, which adapts hidden states rather than model weights. Parallel Control (Zhang et al., 2025) applies low-rank controls to hidden states, reducing training memory by avoiding intermediate-state storage, while Mixture-of-Control (MoC) (Nguyen et al., 2026) shares control information across layers to improve memory efficiency and adaptation quality. However, state controls cannot be merged into the pretrained weights and must be evaluated during every forward pass. Existing methods typically intervene after each attention and feed-forward subblock, causing inference overhead to scale with model depth. Moreover, MoC routes complete low-rank controls, forcing their constituent features to share one routing decision and limiting adaptation flexibility.

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

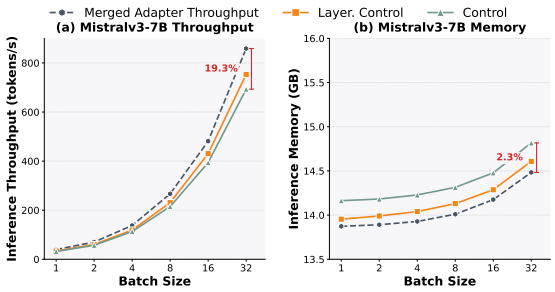

Figure 1: Inference throughput and memory comparison on Mistralv3-7B-Instruct across control granularities and the merged adapter.

Table 1: Average commonsense reasoning accuracy of subblock-level and Layerwise Control across four LLM backbones.

Model	Control	Layer. Control
LLaMA2-7B	79.6	**79.8**
Mistralv3-7B	83.1	**84.5**
Gemma3-12B	83.7	**85.7**
Qwen2.5-14B	87.2	**89.1**

Together, these observations suggest that efficient state-based adaptation requires jointly reconsidering where state interventions are evaluated and how adaptation capacity is communicated across the remaining intervention sites. This distinguishes the problem from weight-based parameter sharing or feature routing. Because state-dependent controls cannot be merged into the pretrained weights, their placement directly determines inference overhead, raising the following research questions:

($\mathcal{RQ}1$) How can the inference throughput of existing state-based adaptation methods be improved while preserving their memory efficiency?

($\mathcal{RQ}2$) How should inter-layer communication be formulated to improve efficiency without compromising performance?

To address these questions, we introduce SURF (**S**imilarity-Guided **U**nified **R**outing of **F**eatures). SURF first replaces subblock-level discretization with coarser layer-wise transitions, directly reducing the frequency of control evaluation. It then measures hidden-state similarity in the pretrained model and groups consecutive layers with similar representations. Within each group, SURF evaluates a control only once and applies one intervention at the coarse-transition boundary. A sparse token-dependent router selects the rank-one features from a shared pool, enabling model-aware sharing features across depth with input-dependent adaptation at the token level.

Our contributions are summarized as follows:

- We identify frequent control evaluations and control-level routing as two key bottlenecks in state-based methods. We then introduce **SURF**, a unified state-based adaptation framework that integrates coarse-grained control placement with similarity-guided layer grouping, and sparse token-dependent routing over shared rank-one control features.
- We theoretically characterize the intervention–capacity trade-off induced by coarse state-control discretization and establish when shared feature compression necessarily induces cross-group reuse.
- Across 10 pretrained architectures on 34 datasets spanning language, reasoning, vision, and time-series benchmarks, SURF achieves a favorable accuracy–efficiency trade-off, matching or outperforming existing state-based methods while improving the inference throughput and preserving training-memory efficiency.

2 Motivation

We identify two sources of avoidable overhead of state-based fine-tuning (Zhang et al., 2024b): frequent control interventions and coarse adaptation sharing.

2.1 Discretization as the Inference Bottleneck in State-Based Adaptation

Consider hidden states $\boldsymbol{X} \in \mathbb{R}^{B\times m_X\times d}$, where B, m_X, and d denote the batch size, sequence length, and hidden dimension, respectively. An L-layer Transformer has the total pretrained forward cost of $\mathcal{O}\left(LBm_X^2d + LBm_Xd^2 + LBm_Xdd_{\text{ff}}\right)$. A rank-$r$ state control $g(\boldsymbol{X})$ adds only $\mathcal{O}(SBm_Xdr)$ across S intervention sites, which appears negligible since $r \ll d$. Nevertheless,

subblock-level control incurs substantial throughput overhead. In Figure 1, it intervenes after both attention and FFN modules ($S = 2L$) at batch size 32, reducing inference throughput by approximately 20% relative to the merged weight-based baseline, while increasing peak inference memory by only 2.3%. The widening gap at larger batch sizes suggests that repeated control operations and activation movement, rather than FLOPs and peak memory alone, drive the slowdown. This raises a central question: *"Do state-based methods intervene more frequently than necessary?"*

From subblock-level to layer-wise control. We treat each Transformer layer ℓ as a single frozen state transition with one layer-wise control:

$$\boldsymbol{z}_\ell = \Phi_\ell(\boldsymbol{z}_{\ell-1}) + g_\ell(\boldsymbol{z}_{\ell-1}), \tag{1}$$

where $\Phi_\ell = \boldsymbol{z}_{\ell-1} + \mathrm{Attn}_\ell(\boldsymbol{z}_{\ell-1}) + \mathrm{FFN}_\ell(\boldsymbol{z}_{\ell-1} + \mathrm{Attn}_\ell(\boldsymbol{z}_{\ell-1}))$ is the complete frozen transition, and g_ℓ is its control (normalization operators and architecture-specific details are omitted for simplicity). This coarsening reduces the intervention count from approximately $2L$ to L without modifying the frozen backbone. Under matched parameter budgets on 32-layer Mistralv3-7B-Instruct, rank-128 Layer-wise Control consistently narrows the throughput gap to the merged baseline, retaining approximately 86–89% of its throughput, compared with about 80% for rank-64 subblock-level Control (Figure 1(a)), while the remaining comparable peak inference memory to the merged baseline and improving downstream accuracy (Figure 1(b), Table 1), indicating that subblock-level interventions may be unnecessarily frequent.

2.2 Rethinking the Granularity of Control Communication

Coarser discretization reduces state-based control evaluations but raises a complementary question: *"How should adaptation information be shared across the remaining control sites?"*. Independent control limits adaptation capacity to individual depths, whereas cross-control communication allows useful task-specific information to be reused across states. Recent state-based approaches such as MoC (Nguyen et al., 2026) support such sharing by routing among low-rank controllers.

Given an input $\boldsymbol{x} \in \mathbb{R}^d$, a rank-$r$ controller $g : \mathbb{R}^d \to \mathbb{R}^d$ can be decomposed as

$$g(\boldsymbol{x}) = \boldsymbol{B}\boldsymbol{A}\boldsymbol{x} = \sum_{q=1}^{r} \boldsymbol{b}_q \boldsymbol{a}_q^\top \boldsymbol{x}, \tag{2}$$

where $\boldsymbol{a}_q$ and $\boldsymbol{b}_q$ denote the q-th columns of $\boldsymbol{A}^\top$ and $\boldsymbol{B}$, respectively. Routing the entire controller therefore couples these components to the same communication decision, potentially limiting adaptation information reuse across control sites.

These observations motivate SURF's two components: similarity-guided grouping to reduce control evaluations; and sparse, token-dependent routing over a shared pool of rank-one features.

3 Methodology

In this section, we develop a unified framework that addresses the two research questions mentioned in Section 1, unifies existing state-based adaptation methods, and provides the theoretical basis. Detailed proofs are provided in Appendix, Section C.

We introduce SURF, a model-aware framework consisting of two stages: **Stage 1** groups layers by their pretrained model similarity, and **Stage 2** routes each group over a unified feature pool to incorporate cross-group sharing with group-specific adaptation (see Figure 2 and Appendix Algorithm 1 for more details).

3.1 Stage 1: Similarity-Guided Layerwise Grouping

Stage 1 of SURF reduces the frequency of state-dependent interventions by grouping consecutive Transformer layers into coarser state transitions, fostering a more efficient process. To avoid task-specific boundary search during fine-tuning, we use pretrained similarity as an efficient, model-dependent criterion for layer grouping. Comparisons with alternative grouping schemes are provided in Section 4.3.

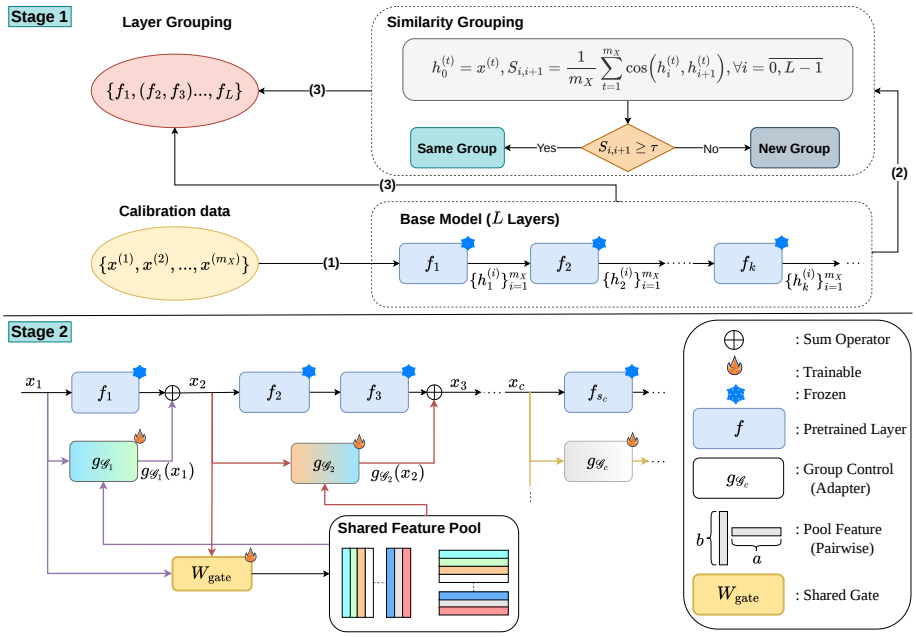

Figure 2: **SURF Architecture.** **Stage 1** groups consecutive Transformer layers according to their pretrained hidden-state similarity and replaces fine-grained interventions with a single control evaluated at each group boundary. **Stage 2** equips these group-level controls with a shared pool of rank-one adaptation features, where a sparse token-dependent router selects the features required for each group-entry state.

Population-level similarity and fixed grouping. Given an input sequence $\boldsymbol{X}$ of length $m_{\boldsymbol{X}}$. Let $\boldsymbol{h}_i^{(0)}(\boldsymbol{X}) = X$, $\boldsymbol{h}_i^{(t)}(\boldsymbol{X}) \in \mathbb{R}^d$ denote the representation of token t after pretrained layer i, and $\hat{\boldsymbol{h}}_i^{(t)}(\boldsymbol{X}) = \frac{\boldsymbol{h}_i^{(t)}(\boldsymbol{X})}{\|\boldsymbol{h}_i^{(t)}(\boldsymbol{X})\|_2}$. The population similarity between layers i and j is defined as

$$S_{i,j} = \mathbb{E}_{\boldsymbol{X}\sim\mathcal{D}}\left[\frac{1}{m_{\boldsymbol{X}}}\sum_{t=1}^{m_{\boldsymbol{X}}}\left\langle \hat{\boldsymbol{h}}_i^{(t)}(\boldsymbol{X}), \hat{\boldsymbol{h}}_j^{(t)}(\boldsymbol{X})\right\rangle\right]. \tag{3}$$

SURF estimates Equation (3) once using a fixed calibration set $\mathcal{D}_{\text{cal}} \subset \mathcal{D}$, and keeps the resulting grouping unchanged throughout fine-tuning and inference. For each task, $\mathcal{D}_{\text{cal}}$ contains 128 examples sampled once and held fixed across all experimental settings.

Given the similarity matrix $\boldsymbol{S}$, each contiguous group $\mathcal{G}_c = \{s_c, s_c+1, \ldots, e_c\}$ is constructed by

$$S_{j,j+1} \geq \tau, \qquad \forall j \in \{s_c, \ldots, e_c - 1\}. \tag{4}$$

A new group starts when this condition is violated. Higher τ yields more conservative groups and more frequent interventions, whereas lower τ permits larger groups and fewer control evaluations.

Geometric interpretation. From Equations (3) and (4), it follows that

$$\mathbb{E}_{\boldsymbol{X}\sim\mathcal{D}}\left[\frac{1}{m_{\boldsymbol{X}}}\sum_{t=1}^{m_{\boldsymbol{X}}}\left\|\hat{\boldsymbol{h}}_j^{(t)}(\boldsymbol{X}) - \hat{\boldsymbol{h}}_{j+1}^{(t)}(\boldsymbol{X})\right\|_2^2\right] = 2(1 - S_{j,j+1}) \leq 2(1-\tau).$$

Thus, τ directly bounds the average normalized representation drift between adjacent layers retained within a group, thereby controlling the coarsening granularity.

One control per coarse transition. For each group $\mathcal{G}_c = \{s_c, \ldots, e_c\}$, define the frozen c-th group transition $F_c = \Phi_{e_c} \circ \cdots \circ \Phi_{s_c}$. Given its group-entry state $\boldsymbol{x}_c$, SURF performs

$$\boldsymbol{x}_{c+1} = F_c(\boldsymbol{x}_c) + g_{\mathcal{G}_c}(\boldsymbol{x}_c), \tag{5}$$

where $g_{\mathcal{G}_c}$ is evaluated once at the group boundary. Thus, a group containing $|\mathcal{G}_c|$ layers requires only one state-dependent control evaluation rather than one intervention at every internal transition.

Intervention–capacity trade-off. We characterize the capacity required for a single group-boundary control to represent the effect of multiple fine-grained interventions. Following prior state-based methods (Zhang et al., 2025; Nguyen et al., 2026), we consider low-rank controls for parameter efficiency.

Theorem 3.1 (Coarse-Control Expressivity via Capacity Aggregation)**.** *Consider a group $\mathcal{G}_c = \{s_c, \dots, e_c\}$ with $M_c := |\mathcal{G}_c|$ and fine-grained controlled dynamics*

$$\boldsymbol{z}_\ell = \Phi_\ell(\boldsymbol{z}_{\ell-1}) + g_\ell(\boldsymbol{z}_{\ell-1}), \qquad g_\ell = \boldsymbol{B}_\ell \psi_\ell, \quad \boldsymbol{B}_\ell \in \mathbb{R}^{d \times r_\ell}, \quad \ell \in \mathcal{G}_c, \tag{6}$$

initialized at $\boldsymbol{z}_{s_c-1} = \boldsymbol{x}_c$, and the frozen group transition $F_c = \Phi_{e_c} \circ \cdots \circ \Phi_{s_c}$. We compare fine-grained controls with a single group-boundary one.

(i) Linear case. *If every Φ_ℓ and ψ_ℓ is linear, there exists a single linear group-boundary control $g_{\mathcal{G}_c}$ with $\mathrm{rank}(g_{\mathcal{G}_c}) \le \sum_{\ell=s_c}^{e_c} r_\ell$ such that $\boldsymbol{z}_{e_c} = F_c(\boldsymbol{x}_c) + g_{\mathcal{G}_c}(\boldsymbol{x}_c) = \boldsymbol{x}_{c+1}$ for every $\boldsymbol{x}_c$.*

(ii) Nonlinear case. *Let $\boldsymbol{x}_\ell^0$ denote the frozen trajectory from $\boldsymbol{x}_c$. Assume that on a compact region around it, each Φ_ℓ is C^1 with bounded, Lipschitz Jacobian and each control g_ℓ is L_g^ℓ-Lipschitz, let $\gamma := \max_{\ell \in \mathcal{G}_c}\{\|g_\ell(\boldsymbol{x}_{\ell-1}^0)\|,\ L_g^\ell\}$ be the* control strength*. Assume that γ is sufficiently small, with $J_\ell := D\Phi_\ell(\boldsymbol{x}_{\ell-1}^0)$ and $J_{j:i} := J_j \cdots J_i$, the propagated group-boundary control*

$$g_{\mathcal{G}_c}(\boldsymbol{x}_c) := \sum_{\ell=s_c}^{e_c} J_{e_c:\ell+1}\, \boldsymbol{B}_\ell \psi_\ell(\boldsymbol{x}_{\ell-1}^0), \qquad \boldsymbol{x}_{c+1} := F_c(\boldsymbol{x}_c) + g_{\mathcal{G}_c}(\boldsymbol{x}_c) \tag{7}$$

lies in a subspace of dimension at most $\sum_{\ell=s_c}^{e_c} r_\ell$, and for every threshold $\gamma_0 > 0$ there is a constant $C_c = C_c(M_c, \Lambda, H, \gamma_0)$, independent of γ and $\boldsymbol{x}_c$, such that

$$\left\|\boldsymbol{z}_{e_c} - \boldsymbol{x}_{c+1}\right\|_2 \ \le\ C_c\, \gamma^2, \quad \text{for all } \gamma \le \gamma_0, \tag{8}$$

where Λ and H bound the Jacobian and its Lipschitz modulus. Moreover, $C_c = 0$ when $M_c = 1$, and $\|\boldsymbol{z}_{e_c} - \boldsymbol{x}_{c+1}\|_2 = O(\gamma^2) \to 0$ as $\gamma \to 0$.

Theorem 3.1 exposes an intervention-capacity trade-off: multiple low-rank interventions are consolidated exactly in the linear case, and in the nonlinear case up to an error quadratic in the control strength γ, while the required boundary capacity grows with their aggregate rank.

3.2 Stage 2: Mixture of Features (MoF) for Group-Level Routing

Stage 2 routes each group over a shared pool of rank-one features, enabling cross-depth reuse and state-dependent specialization. Let $\boldsymbol{A} \in \mathbb{R}^{R \times d}$ and $\boldsymbol{B} \in \mathbb{R}^{d \times R}$ contain R shared rank-one feature pairs $\{(a_q, b_q)\}_{q=1}^R$, and let $\boldsymbol{W}_{\text{gate}} \in \mathbb{R}^{R \times d}$ be the shared router. For each token state $\boldsymbol{x}_c \in \mathbb{R}^d$ entering $\mathcal{G}_c$, with feature budget $k_c \le R$, **Mixture-of-Features (MoF)** computes

$$g_{\mathcal{G}_c}(\boldsymbol{x}_c) = \alpha \boldsymbol{B}\left(\rho_c(\boldsymbol{x}_c) \odot \boldsymbol{A}\boldsymbol{x}_c\right) = \alpha \sum_{q=1}^{R} \rho_{c,q}(\boldsymbol{x}_c)\, \boldsymbol{b}_q \boldsymbol{a}_q^\top \boldsymbol{x}_c, \tag{9}$$

where α are scaling hyperparameter and $\rho_{c,q}(\boldsymbol{x}_c) = \mathrm{softmax}\left(\mathrm{TopK}\left(\boldsymbol{W}_{\text{gate}}\boldsymbol{x}_c, k_c\right)\right)_q \in \mathbb{R}$. For simplicity, we set α default as 1 in all of our experiments, $\mathrm{TopK}(\boldsymbol{u}, k_c)$ retains the k_c largest entries of $\boldsymbol{u}$ and masks all others to $-\infty$ before the softmax, so that $\rho_{c,q}(\boldsymbol{x}_c)$ has exactly k_c nonzero entries. Both the feature pool and router are shared across groups, while the routing pattern and feature budget vary across group-entry states.

Computational complexity. MoF computes routing scores over R shared features, requiring $\mathcal{O}(Bm_X dR)$ computation per group boundary. Evaluating the Top-k selected rank-one features adds $\mathcal{O}(Bm_X dk_c)$, yielding a total control complexity of $\mathcal{O}(Bm_X d(CR + \sum_c k_c))$ across C groups. Although MoF introduces a higher per-boundary cost than a conventional low-rank control, SURF uses substantially fewer intervention boundaries ($C \ll 2L$); we evaluate the resulting throughput in Section 4.

Number of Features per Group. Let $k_c \leq R$ denote the number of active features allocated to group $\mathcal{G}_c$. We consider two schemes: (1) **Fixed**, which sets $k_c = k$ for all groups, and (2) **Adaptive**, with $k_c = \min(k|\mathcal{G}_c|, R)$. Motivated by Theorem 3.1, the adaptive scheme allocates more features to coarser transitions that replace more intervention sites, whereas the fixed scheme uses a uniform feature budget across groups. We compare both schemes in Section 4.3.

SURF reduces control computation through fewer interventions and a compact shared feature pool, while feature-level routing enables state-dependent reuse across groups. A detailed comparison between SURF and existing state-based control frameworks is provided in Appendix Section D. We next establish when compression necessitates such reuse.

Proposition 3.2 (Compression-Induced Feature Reuse)**.** *Consider $C \geq 2$ control groups sharing a pool of R features. For a fixed routed token trajectory $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_C)$, let $\mathcal{S}_c := \mathrm{supp}\big(\rho_c(\boldsymbol{x}_c)\big) \subseteq [R]$, where $[R] := \{1, 2, \ldots, R\}$ denote the active feature set of group c with and $|\mathcal{S}_c| = k_c$. Define the aggregate feature demand $K := \sum_{c=1}^{C} k_c$. Then we obtain*

$$\sum_{m<n} |\mathcal{S}_m \cap \mathcal{S}_n| \geq \frac{K}{2}\left(\frac{K}{R} - 1\right). \tag{10}$$

Consequently, if $R < K$, there is at least one feature reused across multiple groups. Moreover, letting $\bar{k} := K/C$ denote the average feature budget, then

$$\frac{1}{\binom{C}{2}} \sum_{m<n} \frac{|\mathcal{S}_m \cap \mathcal{S}_n|}{\bar{k}} \geq \frac{K/R - 1}{C - 1}. \tag{11}$$

Proposition 3.2 shows that cross-group feature reuse is necessary when the aggregate feature demand exceeds the shared pool size ($K > R$); otherwise, reuse is learned rather than required. Equation (11) quantifies budget-normalized overlap, and Section E.3 reports frequency-based reuse metrics under token-dependent routing.

4 Experiments

This section describes the experimental setup, reports the main results compared to Weight-FT and State-based baselines, and presents ablation studies. Additional results are provided in the Appendix.

4.1 Experimental Settings

We evaluate SURF on eight commonsense-reasoning datasets using LLaMA2-7B (Touvron et al., 2023), Mistral-7B-Instruct (Jiang et al., 2023), Gemma3-12B-Instruct (Team et al., 2025), and Qwen2.5-14B-Instruct (Qwen et al., 2025); 19 VTAB-1K (Zhai et al., 2019) tasks using ViT-Tiny and ViT-Base; three mathematical reasoning benchmarks using Qwen2.5-7B-Instruct (Qwen et al., 2025), Qwen3-30B-A3B-Instruct (Yang et al., 2025); and four long-term forecasting datasets (Zhou et al., 2021; Li et al., 2019) using GPT2 and GPT2-Large (Radford et al., 2019). For simplicity, for instruction-tuned models, we use names with and without the "Instruct" suffix interchangeably throughout the paper. We compare our SURF with LoRA (Hu et al., 2022), DoRA (Liu et al., 2024), MoDE (Ning et al., 2025), MoLEx (Teo & Nguyen, 2025), Parallel Control (Zhang et al., 2025) (abbreviated as Control) and the recent MoC (Nguyen et al., 2026). For commonsense reasoning, vision, and mathematical reasoning tasks, we report accuracy across the eight reasoning benchmarks and 19 VTAB-1K datasets, respectively. For time-series forecasting, we evaluate performance using mean squared error (MSE) and mean absolute error (MAE), following standard forecasting protocols. Detailed datasets, models, and training configurations are provided in Section E.1.

4.2 Main Results

We use LoRA as the reference baseline; $\uparrow$ and $\downarrow$ denote improvement and degradation, respectively. Unless stated otherwise, we use the parameter-matched setting of SURF in subsequent experiments.

Language Tasks. Table 2 shows that SURF achieves the best average performance across four LLM backbones and eight commonsense reasoning benchmarks. Under comparable parameter budgets, it outperforms LoRA by 1.4%–9.1% and the strongest baseline, MoC, by 0.7%–1.2%. On

Table 2: Comparison on commonsense reasoning benchmarks. * denotes previously reported results; best results are in **bold**. SURF is reported under reduced and matched parameter budgets.

Model	Method	% Params	BoolQ	PIQA	SIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Avg (↑)	Time (↓) (h)	GPU Mem. (↓) (GB)
LLaMA2-7B	LoRA	0.52	70.9	82.6	79.4	92.3	81.9	82.8	67.3	83.2	80.0	5.5	47.5
	DoRA	0.53	71.7↑0.8	82.8↑0.2	79.7↑0.3	92.4↑0.1	83.3↑1.4	82.8↑0.0	66.8↓0.5	79.0↓4.2	79.8↓0.2	10.7	69.2
	MoDE	0.53	70.4↓0.5	81.6↓1.0	79.5↑0.1	93.4↑1.1	81.9↑0.0	83.5↑0.7	69.5↑2.2	82.2↓1.0	80.2↑0.2	8.7	56.6
	MoLEx	0.53	70.3↓0.6	82.6↑0.0	79.3↓0.1	89.9↓2.4	78.5↓3.4	84.1↑1.3	70.0↑2.7	83.0↓0.2	79.7↓0.3	11.0	76.9
	Control*	0.50	72.3↑1.4	82.5↓0.1	79.2↓0.2	89.1↓3.2	83.1↑1.2	83.0↑0.2	68.5↑1.2	79.0↓4.2	79.6↓0.4	4.0	42.6
	MoC*	0.51	71.8↑0.9	83.2↑0.6	79.6↑0.2	93.3↑1.0	**84.2**↑2.3	83.6↑0.8	68.6↑1.3	80.4↓2.8	80.5↑0.5	5.5	47.6
	SURF	0.09	71.5↑0.6	81.6↓1.0	**80.3**↑0.9	93.2↑0.9	83.3↑1.4	84.6↑1.8	70.2↑2.9	81.6↓1.6	80.8↑0.8	3.3	45.5
	SURF	0.50	**72.5**↑1.6	**83.7**↑1.1	80.0↑0.6	**93.6**↑1.3	80.8↓1.1	**85.5**↑2.7	**72.0**↑4.7	**83.4**↑0.2	**81.4**↑1.4	5.3	47.2
Mistralv3-7B	LoRA	0.49	67.8	83.0	77.8	89.4	79.3	78.7	64.9	79.2	77.5	5.8	54.3
	DoRA	0.50	68.1↑0.3	82.4↓0.6	77.7↓0.1	88.1↓1.3	78.7↓0.6	77.1↓1.6	64.3↓0.6	79.4↑0.2	77.0↓0.5	9.5	64.7
	MoDE	0.50	72.6↑4.8	87.8↑4.8	80.0↑2.2	95.3↑5.9	86.2↑6.9	87.3↑8.6	77.5↑12.6	86.4↑7.2	84.1↑6.6	7.1	63.5
	MoLEx	0.49	69.5↑1.7	83.5↑0.5	77.9↑0.1	89.4↑0.0	82.2↑2.9	80.1↑1.4	67.6↑2.7	81.0↑1.8	78.9↑1.4	10.0	75.4
	Control*	0.46	70.6↑2.8	86.9↑3.9	79.0↑1.2	94.2↑4.8	84.8↑5.5	87.9↑9.2	76.5↑11.6	84.8↑5.6	83.1↑5.6	4.0	53.2
	MoC*	0.48	74.2↑6.4	88.9↑5.9	81.0↑3.2	95.6↑6.2	**87.9**↑8.6	88.9↑10.2	79.5↑14.6	87.8↑8.6	85.5↑8.0	6.0	55.6
	SURF	0.23	73.8↑6.0	**90.7**↑7.7	81.4↑3.6	**96.0**↑6.6	86.9↑7.6	91.0↑12.3	81.1↑16.2	89.2↑10.0	86.3↑8.8	3.9	52.1
	SURF	0.46	**74.9**↑7.1	90.1↑7.1	**81.9**↑4.1	**96.0**↑6.6	87.0↑7.7	**91.5**↑12.8	**81.3**↑16.4	**90.2**↑11.0	**86.6**↑9.1	6.0	54.9
Gemma3-12B	LoRA	0.44	72.8	89.3	81.1	94.8	86.7	91.3	78.8	88.2	85.4	10.5	69.5
	DoRA	0.45	70.5↓2.3	89.1↓0.2	80.5↓0.6	94.9↑0.1	**87.8**↑1.1	91.0↓0.3	81.2↑2.4	88.6↑0.4	85.4↑0.0	19.7	114.5
	MoDE	0.41	72.8↑0.0	88.0↓1.3	78.1↓3.0	91.2↓3.6	83.0↓3.7	95.8↑4.5	88.3↑9.5	89.2↑1.0	85.8↑0.4	11.8	101.5
	MoLEx	0.45	72.4↓0.4	88.7↓0.6	80.9↓0.2	**95.3**↑0.5	86.7↑0.0	91.5↑0.2	81.3↑2.5	88.8↑0.6	85.7↑0.3	13.0	117.8
	Control	0.39	72.6↓0.2	86.1↓3.2	79.6↓1.5	93.2↓1.6	85.3↓1.4	88.8↓2.5	77.7↓1.1	86.0↓2.2	83.7↓1.7	8.3	66.7
	MoC	0.40	71.3↓1.5	87.6↓1.7	**82.0**↑0.9	95.1↑0.3	86.7↑0.0	91.5↑0.2	83.2↑4.4	88.4↑0.2	85.7↑0.3	11.7	70.0
	SURF	0.13	**75.0**↑2.2	**89.9**↑0.6	79.6↓1.5	93.8↓1.0	84.1↓2.6	**96.0**↑4.7	87.2↑8.4	89.6↑1.4	**86.9**↑1.5	7.9	67.6
	SURF	0.39	74.5↑1.7	89.4↑0.1	78.9↓2.2	92.8↓2.0	83.6↓3.1	95.8↑4.5	**89.2**↑10.4	**89.8**↑1.6	86.8↑1.4	8.5	69.8
Qwen2.5-14B	LoRA	0.43	72.9	88.0	80.7	94.4	87.3	92.8	85.8	93.6	86.9	7.9	60.9
	DoRA	0.44	73.8↑0.9	88.8↑0.8	81.5↑0.8	95.6↑1.2	88.2↑0.9	94.3↑1.5	86.6↑0.8	89.0↓4.6	87.2↑0.3	14.0	90.4
	MoDE	0.43	74.6↑1.7	92.0↑4.0	82.3↑1.6	96.2↑1.8	87.8↑0.5	97.5↑4.7	93.2↑7.4	94.2↑0.6	89.7↑2.8	14.3	97.8
	MoLEx	0.43	73.4↑0.5	89.8↑1.8	81.7↑1.0	95.3↑0.9	88.2↑0.9	94.1↑1.3	86.7↑0.9	91.2↓2.4	87.6↑0.7	19.4	116.3
	Control*	0.43	72.8↓0.1	88.8↑0.8	81.3↑0.6	94.6↑0.2	87.5↑0.2	94.4↑1.6	86.9↑1.1	91.2↓2.4	87.2↑0.3	6.5	58.7
	MoC*	0.43	75.5↑2.6	91.8↑3.8	82.4↑1.7	**96.8**↑2.4	**91.0**↑3.7	96.6↑3.8	90.2↑4.4	96.4↑2.8	90.1↑3.2	8.0	61.3
	SURF	0.14	**76.8**↑3.9	**93.3**↑5.3	82.4↑1.7	96.5↑2.1	90.3↑3.0	97.3↑4.5	92.7↑6.9	**96.8**↑3.2	**90.8**↑3.9	6.2	60.4
	SURF	0.43	76.0↑3.1	93.0↑5.0	**83.2**↑2.5	96.6↑2.2	89.7↑2.4	**97.7**↑4.9	**93.4**↑7.6	**96.8**↑3.2	**90.8**↑3.9	6.6	62.0

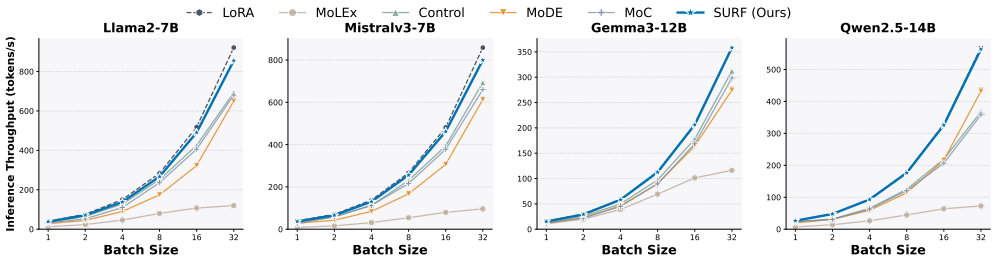

Figure 3: Inference Throughput of SURF compared to prior methods across LLMs and batch size.

Gemma3-12B-Instruct and Qwen2.5-14B-Instruct, SURF achieves comparable averages (86.9 and 90.8) with approximately 3× fewer trainable parameters, highlighting the benefits of feature-level capacity reuse. SURF also improves inference efficiency while remaining competitive in training cost: grouping increases throughput over state-based methods, especially at scale (Figure 3), and single-token decoding latency remains close to the merged weight-based baseline (Section E.3). Overall, SURF balances adaptation quality, parameter efficiency, and computational efficiency.

Table 3: Comparison on three mathematical reasoning benchmarks.

Model	Method	MATH-500	AMC23	GSM8K	Avg. (↑)
Qwen2.5-7B -Instruct	LoRA	**51.2**	40.0	87.5	59.6
	Control	47.2↓4.0	37.5↓2.5	84.3↓3.2	56.3↓3.3
	MoC	**51.2**↑0.0	42.5↑2.5	86.7↓0.8	60.1↑0.5
	SURF	**51.2**↑0.0	**50.0**↑10.0	**88.0**↑0.5	**63.1**↑3.5
Qwen3-30B-A3B-Instruct	LoRA	62.8	57.5	94.0	71.4
	Control	60.8↓2.0	55.0↓2.5	91.5↓2.5	69.1↓2.3
	MoC	60.8↓2.0	70.0↑12.5	91.5↓2.5	74.1↑2.7
	SURF	**63.6**↑0.8	**77.5**↑20.0	**94.3**↑0.3	**78.5**↑7.1

Table 4: Comparison on Long-Term Time-Series Forecasting (MSE/MAE).

Model	Method	ETTh1	ETTm1	ECL	Weather	Avg (↓)
GPT2	LoRA	0.443/0.441	0.356/0.386	0.165/0.258	0.231/0.268	0.299/0.338
	Control	0.441/0.437	0.357/0.387	0.163/0.257	0.231/0.269	0.298/0.337
	MoC	0.434/0.434	0.358/0.385	0.165/0.258	0.244/0.276	0.300/0.338
	SURF	**0.417/0.425**	**0.345/0.374**	**0.162/0.255**	**0.226/0.264**	**0.287/0.329**
GPT2-Large	LoRA	0.457/0.453	0.379/0.401	0.167/0.262	0.275/0.294	0.319/0.353
	Control	0.446/0.444	0.380/0.401	0.165/0.260	0.261/0.289	0.313/0.349
	MoC	0.469/0.461	0.375/0.399	0.169/0.264	0.256/0.288	0.317/0.353
	SURF	**0.427/0.429**	**0.360/0.390**	**0.160/0.255**	**0.230/0.268**	**0.294/0.335**

Reasoning Tasks. We next compare SURF against LoRA, Control, and MoC, which provide competitive performance–efficiency trade-offs in Table 2 on more challenging mathematical reasoning benchmarks. Across MATH-500, AMC23, and GSM8K, SURF achieves the highest average accuracy on both Qwen2.5-7B-Instruct and Qwen3-30B-A3B-Instruct (Table 3), outperforming LoRA

by approximately 3.5% and 7.1%, and the strongest baseline by approximately 3.0% and 4.4%, respectively. These gains highlight the benefits of state communication, feature-level sharing, and similarity-guided grouping for reasoning.

Time-Series Tasks. We compare SURF against LoRA, Control, and MoC on long-term time-series forecasting across four datasets and two pretrained models. As shown in Table 4, SURF consistently achieves lower MSE and MAE than all baselines, further demonstrating its effectiveness as a general state-based adaptation framework across diverse modalities and task structures.

Vision Tasks. To assess generalization beyond language, we evaluate SURF on VTAB-1K using ViT-Tiny and ViT-Base under matched parameter budgets. SURF achieves the highest average accuracy across 19 Natural, Specialized, and Structured datasets (Table 5). It outperforms LoRA by 2.4% and 1.6%, and the strongest baseline by 2.1% and 1.4% on ViT-Tiny and ViT-Base, respectively. SURF also shows competitive training time and memory usage compared with LoRA, Control, and MoC, highlighting its advantages in performance, time, and memory efficiency. Detailed results are provided in the Appendix; (Figure 6 and Table 16).

Table 5: Comparison on VTAB-style benchmarks.

Model	Method	Natural (↑)	Specialized (↑)	Structured (↑)	Avg (↑)	Time (m) (↓)	GPU Mem. (GB) (↓)
ViT-Tiny	LoRA	65.8	84.4	47.3	61.9	1.2	4.2
	DoRA	66.3↑0.5	85.1↑0.7	46.4↓0.9	61.9↑0.0	1.6↓0.4	8.1↓3.9
	MoDE	66.9↑1.1	83.4↓1.0	47.4↑0.1	62.2↑0.3	1.5↓0.3	7.5↓3.3
	MoLEx	66.1↑0.3	**85.5**↑1.1	46.3↓1.0	61.8↓0.1	2.4↓1.2	10.6↓6.4
	Control	66.1↑0.3	84.6↑0.2	45.8↓1.5	61.5↓0.4	0.8↑0.4	2.1↑2.1
	MoC	68.5↑2.7	82.6↓1.8	46.0↓1.3	62.0↑0.1	1.1↑0.1	2.4↑1.8
	SURF	**69.4**↑3.6	**85.5**↑1.1	**49.3**↑2.0	**64.3**↑2.4	0.9↑0.3	2.1↑2.1
ViT-Base	LoRA	79.4	87.0	51.9	69.4	2.6	18.6
	DoRA	79.0↓0.4	86.4↓0.6	51.3↓0.6	68.9↓0.5	6.7↓4.1	33.1↓14.5
	MoDE	79.4↑0.0	87.3↑0.3	49.7↓2.2	68.5↓0.9	6.4↓3.8	29.5↓10.9
	MoLEx	79.1↓0.3	87.5↑0.5	49.9↓2.0	68.6↓0.8	8.5↓5.9	36.7↓18.1
	Control	78.8↓0.6	88.1↑1.1	51.1↓0.8	69.1↓0.3	1.9↑0.7	10.8↑7.8
	MoC	80.2↑0.8	87.3↑0.3	51.4↓0.5	69.6↑0.2	2.2↑0.4	10.9↑7.7
	SURF	**80.3**↑0.9	**88.5**↑1.5	**54.1**↑2.2	**71.0**↑1.6	2.0↑0.6	10.8↑7.8

4.3 Ablation Studies

We ablate SURF's sensitivity, grouping and routing strategies, components and allocation schemes, and control types.

Grouping and Routing Schemes. On Mistralv3-7B-Instruct, similarity-guided grouping outperforms layerwise, uniform, and random grouping with only 16 control evaluations (Table 6). For routing, uniform sharing improves over independent layerwise controls, showing that cross-group sharing helps. Dynamic control-level routing yields a further gain, and feature-level MoF performs best, improving the average score by 1.5% over layerwise control and 0.5% over MoC (Table 7).

Components and Allocation. We evaluate SURF's components and its **Fixed** and **Adaptive** schemes on Mistralv3-7B-Instruct and Gemma3-12B-Instruct (Table 14 in Section F), where "–" indicates that the two schemes are equivalent without grouping. Removing either component consistently degrades performance: by about 0.5% on Mistralv3-7B-Instruct, and up to 1.6% and 2.3% on Gemma3-12B-Instruct. Fixed allocation performs better on Mistral, while Adaptive performs better on Gemma, indicating model-dependent preferences.

Feature Budget k, Pool Size R, and Similarity Threshold τ. Across feature budgets $k \in \{8, 16, 32, 64\}$, Qwen2.5-14B-Instruct and Gemma3-12B-Instruct perform strongly at $k = 16$, Mistralv3-7B-Instruct at $k = 32$, and LLaMA2-7B at $k = 64$, using far fewer features than fixed allocations of up to 128 per layer and adding modest overhead (Figure 4(a,b) in Section F). Across pool sizes $R \in \{512, 1366, 2731, 4096\}$, strong performance persists at smaller pools: LLaMA2-7B and Gemma3-12B-Instruct remain competitive at $R = 1366$ with only 33%–50% of the trainable parameters, while Mistralv3-7B-Instruct and Qwen2.5-14B-Instruct peak at $R = 2731$ and 4096, respectively (Figure 4(c)). Cross-group reuse remains consistent across k and R (Figures 7 to 15 in Section E.3). Finally, thresholds $\tau \in [0.90, 0.99]$ generally perform best, reducing controls from 64–96 to 6–28, an 8–16× reduction, without sacrificing adaptation quality (Figure 4(d)).

Control Types. We consider $g_{\mathcal{G}_c}(\boldsymbol{x}_c) = \alpha\, \phi_2(\boldsymbol{B}\,(\rho_c(\boldsymbol{x}_c) \odot \phi_1(\boldsymbol{A}\boldsymbol{x}_c)))$, fix $\phi_2 = \mathrm{Identity}$ and vary $\phi_1 \in \mathrm{Identity}, \mathrm{SiLU}, \mathrm{Tanh}$, covering both linear and nonlinear controllers. Although nonlinear controllers (SiLU) improve Natural on ViT-Tiny and Specialized on ViT-Base (Table 15), the default setting with $\phi_1 = \phi_2 = \mathrm{Identity}$ achieves the strongest performance with a simpler formulation.

Table 6: Ablation of different control grouping strategies in Mistralv3-7B-Instruct model.

Method	Grouping	# Ctrl. Evals	Avg (↑)
Layer. Control	none	32	84.5
Uniform	uniform contiguous	16	84.8
Random	random contiguous	16	85.1
Similarity-guided	**adaptive**	**16**	**85.8**

Table 7: Ablation of feature sharing and dynamic routing strategies in Mistralv3-7B-Instruct model.

Method	Shared Features	Routing Unit	Dynamic	Avg (↑)
Layer. Control	✗	–	✗	84.5
Uniform Sharing	✓	–	✗	85.1
MoC	✓	control	✓	85.5
MoF	✓	**feature**	✓	**86.0**

5 Related Work

Parameter-Efficient Fine-Tuning (PEFT). As Transformer models scale (Vaswani et al., 2017; Dosovitskiy et al., 2020), full fine-tuning becomes increasingly expensive, motivating parameter-efficient fine-tuning (PEFT). Existing approaches either update selected layers (Oquab et al., 2014; Alain & Bengio, 2018; Lee et al., 2023; Kaplun et al., 2023) or introduce lightweight trainable components while freezing the backbone, such as bottleneck adapters (Houlsby et al., 2019; Mahabadi et al., 2021) and continuous prompts (Lester et al., 2021; Liu et al., 2022; Li & Liang, 2021), which can be sensitive to initialization (Razdaibiedina et al., 2023).

Low-Rank Adaptation (LoRA). (Hu et al., 2022) parameterizes weight updates with low-rank matrices, with extensions based on parameter sharing, quantization, adaptive rank allocation, and magnitude–direction decomposition (Kopiczko et al., 2023; Dettmers et al., 2023; Hyeon-Woo et al., 2023; Liu et al., 2024). LoRA and DoRA can be merged into pretrained weights, incurring negligible inference overhead. In contrast, MoDE (Ning et al., 2025) uses input-dependent routing by sharing one low-rank projection while selecting expert-specific complementary projections, which prevents merging and adds computation at every injection site during inference. Most weight-based methods remain layer-local, limiting information sharing across depth. MoLEx (Teo & Nguyen, 2025) enables cross-layer routing, but requires additional layer evaluations.

Control-Theoretic View and State-Based Tuning. State-based tuning views deep networks as dynamical systems over feature states (Li et al., 2018), motivating studies on continuous-depth dynamics, stability, and optimization (Tabuada & Gharesifard, 2024; Lu et al., 2020; Nguyen et al., 2024; Benning et al., 2019). Zhang et al. (Zhang et al., 2024b) formulate LoRA as a control input to pretrained dynamics, while Parallel Control (Zhang et al., 2025) reduces training memory by bypassing intermediate states. More recently, Nguyen et al. (2026) enable cross-layer communication through shared control information. However, they typically retain greater inference throughput overhead than weight-based PEFT and route entire control representations, limiting adaptation sharing. Our SURF addresses these limitations, improving adaptation while retaining comparable training efficiency and narrowing the inference gap with weight-based methods.

6 Conclusion and Limitations

Conclusion. We introduce SURF, a state-based adaptation framework that jointly improves inference efficiency, training efficiency, and adaptation quality. SURF uses similarity-guided coarsening to reduce non-mergeable control evaluations with sparse, token-dependent routing over shared rank-one features for cross-layer feature reuse. Across 34 datasets and 10 pretrained models, SURF matches or outperforms existing state-based methods while reducing trainable parameters by up to $3\times$, maintaining strong training efficiency, and substantially narrowing the inference-throughput gap with mergeable weight-based methods. These results highlight discretization granularity and feature-level communication as key principles for efficient state-based adaptation.

Limitations. SURF has three principal limitations. First, its model-dependent offline grouping may not transfer uniformly across architectures and tasks; jointly optimizing intervention locations and adaptation parameters may improve robustness. However, it introduces additional optimization complexity (Section G). Second, our analysis is exact for linear controls, but yields only a quadratic error bound for smooth nonlinear dynamics, leaving general trajectory-dependent controllers beyond its scope despite their potential benefits (Table 15 in Section G). Finally, developing an end-to-end framework that jointly learns the grouping structure and group-specific feature budgets remains a critical direction for improving the flexibility, efficiency, and generality of state-based adaptation.

AI USE STATEMENT

In this work, we used generative AI tools to assist in proof reasoning and to co-implement methods. We did not use generative AI tools to generate synthetic datasets, develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments, assist with translation, clean or reformat datasets, support qualitative or thematic data analysis, or interpret results.

Additionally, we used generative AI tools to create or modify scientific figures or images, suggest experimental parameters, summarize or analyze existing literature, and edit the paper to improve readability. We reviewed all AI-assisted work. LLM-generated code was verified and tested for correctness by three authors, and each step of AI-assisted proofs was independently verified by two authors. We take responsibility for the final content of this work, including all text, claims, code, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses publicly available benchmark datasets and pretrained models and does not involve human subjects, personally identifiable information, or the collection of sensitive data. We are not aware of any specific ethical concerns arising from the methodology or experiments presented in this work.

REPRODUCIBILITY STATEMENT

We provide the implementation details, training configurations, hyperparameters, dataset and evaluation protocols, and additional ablations required to reproduce our experiments in the main paper and appendix. Complete derivations and proofs for the theoretical results are included in the appendix, together with detailed descriptions of the SURF grouping and routing procedures. An anonymous implementation with configuration files is provided in the supplementary materials to facilitate reproduction of the reported results.

REFERENCES

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes, 2018. URL https://arxiv.org/abs/1610.01644.

Martin Benning, Elena Celledoni, Matthias J. Ehrhardt, Brynjulf Owren, and Carola-Bibiane Schönlieb. Deep learning as optimal control problems: models and numerical methods, 2019. URL https://arxiv.org/abs/1904.05657.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*, 2019.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

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

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms, 2023. URL https://arxiv.org/abs/2305.14314.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020. URL https://arxiv.org/abs/2010.11929.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp, 2019. URL https://arxiv.org/abs/1902.00751.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

Hugging Face. Open r1: A fully open reproduction of deepseek-r1, January 2025. URL https://github.com/huggingface/open-r1.

Nam Hyeon-Woo, Moon Ye-Bin, and Tae-Hyun Oh. Fedpara: Low-rank hadamard product for communication-efficient federated learning, 2023. URL https://arxiv.org/abs/2108.06098.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL https://arxiv.org/abs/2310.06825.

Gal Kaplun, Andrey Gurevich, Tal Swisa, Mazor David, Shai Shalev-Shwartz, and Eran Malach. Less is more: Selective layer finetuning with subtuning. *arXiv preprint arXiv:2302.06354*, 2023.

Dawid J Kopiczko, Tijmen Blankevoort, and Yuki M Asano. Vera: Vector-based random matrix adaptation. *arXiv preprint arXiv:2310.11454*, 2023.

Yoonho Lee, Annie S. Chen, Fahim Tajwar, Ananya Kumar, Huaxiu Yao, Percy Liang, and Chelsea Finn. Surgical fine-tuning improves adaptation to distribution shifts, 2023. URL https://arxiv.org/abs/2210.11466.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning, 2021. URL https://arxiv.org/abs/2104.08691.

Qianxiao Li, Long Chen, Cheng Tai, and Weinan E. Maximum principle based algorithms for deep learning, 2018. URL https://arxiv.org/abs/1710.09513.

Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in neural information processing systems*, 32, 2019.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation, 2021. URL https://arxiv.org/abs/2101.00190.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. In *Forty-first International Conference on Machine Learning*, 2024.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 61–68, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-short.8. URL https://aclanthology.org/2022.acl-short.8/.

Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations, 2020. URL https://arxiv.org/abs/1710.10121.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. Compacter: Efficient low-rank hypercomplex adapter layers, 2021. URL https://arxiv.org/abs/2106.04647.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*, 2018.

Masaaki Nagahara, Daniel E Quevedo, and Dragan Nešić. Maximum hands-off control: a paradigm of control effort minimization. *IEEE Transactions on Automatic Control*, 61(3):735–747, 2015.

Duc Anh Nguyen, Tien Ngoc Luu, Tung Pham, and Toan Tran. Mixture-of-control: State-aware fine-tuning for transformer-based models. *arXiv preprint arXiv:2606.31397*, 2026.

Tam Nguyen, César A. Uribe, Tan M. Nguyen, and Richard G. Baraniuk. Pidformer: Transformer meets control theory, 2024. URL https://arxiv.org/abs/2402.15989.

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.

Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1717–1724, 2014. doi: 10.1109/CVPR.2014.222.

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

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Anastasia Razdaibiedina, Yuning Mao, Rui Hou, Madian Khabsa, Mike Lewis, and Amjad Almahairi. Progressive prompts: Continual learning for language models, 2023. URL https://arxiv.org/abs/2301.12314.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense reasoning about social interactions. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4463–4473, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1454. URL https://aclanthology.org/D19-1454/.

Paulo Tabuada and Bahman Gharesifard. Universal approximation power of deep residual neural networks via nonlinear control theory, 2024. URL https://arxiv.org/abs/2007.06007.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, Gaël Liu, Francesco Visin, Kathleen Kenealy, Lucas Beyer, Xiaohai Zhai, Anton Tsitsulin, Robert Busa-Fekete, Alex Feng, Noveen Sachdeva, Benjamin Coleman, Yi Gao, Basil Mustafa, Iain Barr, Emilio Parisotto, David Tian, Matan Eyal, Colin Cherry, Jan-Thorsten Peter, Danila Sinopalnikov, Surya Bhupatiraju, Rishabh Agarwal, Mehran Kazemi, Dan Malkin, Ravin Kumar, David Vilar, Idan Brusilovsky, Jiaming Luo, Andreas Steiner, Abe Friesen, Abhanshu Sharma, Abheesht Sharma, Adi Mayrav Gilady, Adrian Goedeckemeyer, Alaa Saade, Alex Feng, Alexander Kolesnikov, Alexei Bendebury, Alvin Abdagic, Amit Vadi, András György, André Susano Pinto, Anil Das, Ankur Bapna, Antoine Miech, Antoine Yang, Antonia Paterson, Ashish Shenoy, Ayan Chakrabarti, Bilal Piot, Bo Wu, Bobak Shahriari, Bryce Petrini, Charlie Chen, Charline Le Lan, Christopher A. Choquette-Choo, CJ Carey, Cormac Brick, Daniel Deutsch, Danielle Eisenbud, Dee Cattle, Derek Cheng, Dimitris Paparas, Divyashree Shivakumar Sreepathihalli, Doug Reid, Dustin Tran, Dustin Zelle, Eric Noland, Erwin Huizenga, Eugene Kharitonov, Frederick Liu, Gagik Amirkhanyan, Glenn Cameron, Hadi Hashemi, Hanna Klimczak-Plucińska, Harman Singh, Harsh Mehta, Harshal Tushar Lehri, Hussein Hazimeh, Ian Ballantyne, Idan Szpektor, Ivan Nardini, Jean Pouget-Abadie, Jetha Chan, Joe Stanton, John Wieting, Jonathan Lai, Jordi Orbay, Joseph Fernandez, Josh Newlan, Ju yeong Ji, Jyotinder Singh, Kat Black, Kathy Yu, Kevin Hui, Kiran Vodrahalli, Klaus Greff, Linhai Qiu, Marcella Valentine, Marina Coelho, Marvin Ritter, Matt Hoffman, Matthew Watson, Mayank Chaturvedi, Michael Moynihan, Min Ma, Nabila Babar, Natasha Noy, Nathan Byrd, Nick Roy, Nikola Momchev, Nilay Chauhan, Noveen Sachdeva, Oskar Bunyan, Pankil Botarda, Paul Caron, Paul Kishan Rubenstein, Phil Culliton, Philipp Schmid, Pier Giuseppe Sessa, Pingmei Xu, Piotr Stanczyk, Pouya Tafti, Rakesh Shivanna, Renjie Wu, Renke Pan, Reza Rokni, Rob Willoughby, Rohith Vallu, Ryan Mullins, Sammy Jerome, Sara Smoot, Sertan Girgin, Shariq Iqbal, Shashir Reddy, Shruti Sheth, Siim Põder, Sijal Bhatnagar, Sindhu Raghuram Panyam, Sivan Eiger, Susan Zhang, Tianqi Liu, Trevor Yacovone, Tyler Liechty, Uday Kalra, Utku Evci, Vedant Misra, Vincent Roseberry, Vlad Feinberg, Vlad Kolesnikov, Woohyun Han, Woosuk Kwon, Xi Chen, Yinlam Chow, Yuvein Zhu, Zichuan Wei, Zoltan Egyed, Victor Cotruta, Minh Giang, Phoebe Kirk, Anand Rao, Kat Black, Nabila Babar, Jessica Lo, Erica Moreira, Luiz Gustavo Martins, Omar Sanseviero, Lucas Gonzalez, Zach Gleicher, Tris Warkentin, Vahab Mirrokni, Evan Senter, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, Yossi Matias, D. Sculley, Slav Petrov, Noah Fiedel, Noam Shazeer, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Jean-Baptiste Alayrac, Rohan Anil, Dmitry, Lepikhin, Sebastian Borgeaud, Olivier Bachem, Armand Joulin, Alek Andreev, Cassidy Hardin, Robert Dadashi, and Léonard Hussenot. Gemma 3 technical report, 2025. URL https://arxiv.org/abs/2503.19786.

Rachel SY Teo and Tan M Nguyen. Molex: Mixture of layer experts for finetuning with sparse upcycling. *arXiv preprint arXiv:2503.11144*, 2025.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Xun Wu, Shaohan Huang, and Furu Wei. Mixture of lora experts. *arXiv preprint arXiv:2404.13628*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui

Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019.

Chi Zhang, Jingpu Cheng, and Qianxiao Li. An optimal control view of lora and binary controller design for vision transformers. In *European Conference on Computer Vision*, pp. 144–160. Springer, 2024a.

Chi Zhang, Cheng Jingpu, Yanyu Xu, and Qianxiao Li. Parameter-efficient fine-tuning with controls. In *Forty-first International Conference on Machine Learning*, 2024b.

Chi Zhang, REN Lianhai, Jingpu Cheng, and Qianxiao Li. From weight-based to state-based fine-tuning: Further memory reduction on lora with parallel control. In *Forty-second International Conference on Machine Learning*, 2025.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 11106–11115, 2021.

Rethinking State-Control Dynamics for Unified and Efficient Transformer Adaptation

Supplementary Material

Appendix Contents

A Notations **16**

B Algorithm **16**

C Theoretical Results **17**

C.1 Proof of Theorem 3.1 17

C.2 Proof of Proposition 3.2 19

D Relation to Existing State-Based Controls **20**

E Experimental Details **21**

E.1 Dataset Details 21

E.2 Model Architecture and Training Configuration 22

E.3 Additional Experiments 24

F Ablation Studies **26**

G Discussion **26**

A NOTATIONS

Table 8: Summary of notation used throughout the paper.

Symbol	Definition	Domain / Shape
Backbone and grouping		
L	Number of frozen Transformer layers.	$\mathbb{N}$
d	Hidden-state dimension.	$\mathbb{N}$
Φ_i	Frozen transition implemented by Transformer layer i.	$\mathbb{R}^d \to \mathbb{R}^d$
$\mathcal{D}_{\text{cal}}$	Fixed calibration set used once to construct layer groups.	–
$\boldsymbol{X},\ m_X$	Calibration input sequence and its number of tokens.	–
$\boldsymbol{h}_i^{(t)}(\boldsymbol{X})$	Pretrained representation of token t after layer i.	$\mathbb{R}^d$
$\hat{\boldsymbol{h}}_i^{(t)}(\boldsymbol{X})$	ℓ_2-normalized version of $\boldsymbol{h}_i^{(t)}(\boldsymbol{X})$.	$\mathbb{R}^d$
S_{ij}	Calibration cosine similarity between layers i and j.	$[-1, 1]$
τ	Similarity threshold used to form contiguous groups.	$[-1, 1]$
$\mathcal{G}_c$	c-th contiguous layer group, $\mathcal{G}_c = \{s_c, \dots, e_c\}$.	–
C	Number of layer groups.	$\mathbb{N}$
s_c, e_c	First (anchor) and last layer of group $\mathcal{G}_c$.	$\{1, \dots, L\}$
F_c	Frozen transition across group $\mathcal{G}_c$, $F_c = \Phi_{e_c} \circ \cdots \circ \Phi_{s_c}$.	$\mathbb{R}^d \to \mathbb{R}^d$
$\boldsymbol{x}_c$	Token state entering group $\mathcal{G}_c$.	$\mathbb{R}^d$
$\boldsymbol{X}_c$	Sequence of token states entering group $\mathcal{G}_c$.	$\mathbb{R}^{m \times d}$
Fine-grained and coarse controls		
$g_i(\boldsymbol{x})$	Fine-grained control at layer i.	$\mathbb{R}^d$
$\boldsymbol{B}_i, \psi_i$	Output basis and coefficient map of a fine-grained control, $g_i(\boldsymbol{x}) = \boldsymbol{B}_i \psi_i(\boldsymbol{x})$.	$\boldsymbol{B}_i \in \mathbb{R}^{d \times r_i}$
r_i	Output rank / feature dimension of control g_i.	$\mathbb{N}$
J_i	Jacobian $D\Phi_i(\boldsymbol{x}_{i-1})$ along the frozen trajectory.	$\mathbb{R}^{d \times d}$
$J_{j:i}$	Jacobian product $J_j J_{j-1} \cdots J_i$; an empty product equals I.	$\mathbb{R}^{d \times d}$
$g_{\mathcal{G}_c}(\boldsymbol{x}_c)$	Single control correction evaluated at the boundary of group $\mathcal{G}_c$.	$\mathbb{R}^d$
M_c	Number of layers in group $\mathcal{G}_c$, $M_c = \lvert\mathcal{G}_c\rvert$.	$\mathbb{N}$
γ	Control strength on $\mathcal{G}_c$, bounding both the magnitude $\lVert g_\ell(\boldsymbol{x}_{\ell-1}^0)\rVert$ and the Lipschitz constant of each g_ℓ.	$\mathbb{R}_+$
$\boldsymbol{x}_\ell^0$	Frozen trajectory from $\boldsymbol{x}_c$, $\boldsymbol{x}_\ell^0 = \Phi_\ell(\boldsymbol{x}_{\ell-1}^0)$.	$\mathbb{R}^d$
Mixture of Features		
R	Number of rank-one features in the shared feature pool.	$\mathbb{N}$
$\boldsymbol{A}, \boldsymbol{B}$	Shared input and output feature matrices.	$\boldsymbol{A} \in \mathbb{R}^{R \times d}$, $\boldsymbol{B} \in \mathbb{R}^{d \times R}$
$\boldsymbol{a}_q, \boldsymbol{b}_q$	Input and output vectors of shared rank-one feature q; row q of $\boldsymbol{A}$ is $\boldsymbol{a}_q^\top$ and column q of $\boldsymbol{B}$ is $\boldsymbol{b}_q$.	$\mathbb{R}^d$
$\boldsymbol{W}_{\text{gate}}$	Shared token-wise routing matrix.	$\mathbb{R}^{R \times d}$
$\rho_c(\boldsymbol{x}_c)$	Sparse routing distribution over the shared feature pool.	$\mathbb{R}^R$
k_c	Number of active features allocated to group $\mathcal{G}_c$.	$\{1, \dots, R\}$
α	Scale applied to the MoF boundary correction.	$\mathbb{R}$
$\mathcal{S}_c$	Active feature set of group c for a fixed routed token trajectory.	$\mathcal{S}_c \subseteq [R]$
K	Aggregate feature demand, $K = \sum_{c=1}^{C} k_c$.	$\mathbb{N}$
$\bar{k}$	Average group feature budget, $\bar{k} = K/C$.	$\mathbb{R}_+$

B ALGORITHM

SURF Algorithm is presented in Algorithm 1.

Algorithm 1: SURF (Similarity-Guided Unified Routing of Features)

Input: Input sequence state $\boldsymbol{X}_1$; frozen layers $\{\Phi_i\}_{i=1}^{L}$; calibration set $\mathcal{D}_{\text{cal}}$; threshold τ; shared features $\boldsymbol{A} \in \mathbb{R}^{R\times d}$ and $\boldsymbol{B} \in \mathbb{R}^{d\times R}$; shared gate $\boldsymbol{W}_{\text{gate}} \in \mathbb{R}^{R\times d}$; feature-budget rule (fixed/ adaptive); scale α.

Output: Adapted hidden state $\boldsymbol{X}_{C+1}$.

Stage 1 (offline preprocessing): Similarity-guided grouping

for $i \leftarrow 1$ **to** L **do**

  Compute $\{\boldsymbol{h}_i^{(t)}(\boldsymbol{X})\}_{\boldsymbol{X}\in\mathcal{D}_{\text{cal}}}$

  $\hat{\boldsymbol{h}}_i^{(t)}(\boldsymbol{X}) \leftarrow \boldsymbol{h}_i^{(t)}(\boldsymbol{X})/\|\boldsymbol{h}_i^{(t)}(\boldsymbol{X})\|_2$

foreach (i, j) *such that* $1 \le i < j \le L$ **do**

  $$S_{i,j} \leftarrow \frac{1}{|\mathcal{D}_{\text{cal}}|}\sum_{\boldsymbol{X}\in\mathcal{D}_{\text{cal}}}\frac{1}{m_{\boldsymbol{X}}}\sum_{t=1}^{m_{\boldsymbol{X}}}\left\langle \hat{\boldsymbol{h}}_i^{(t)}(\boldsymbol{X}), \hat{\boldsymbol{h}}_j^{(t)}(\boldsymbol{X})\right\rangle$$

Construct contiguous groups $\{\mathcal{G}_c\}_{c=1}^{C}$, where $\mathcal{G}_c = \{s_c, \dots, e_c\}$ and $S_{j,j+1} \ge \tau, j \in \mathcal{G}_c \setminus \{e_c\}$;

Assign feature budget k_c to each $\mathcal{G}_c$ according to the fixed or adaptive scheme

Freeze $\{\mathcal{G}_c\}_{c=1}^{C}$ for all subsequent fine-tuning and inference

Stage 2 (fine-tuning and inference): Group-level Mixture of Features

for $c \leftarrow 1$ **to** C **do**

  $\boldsymbol{Y}_c \leftarrow F_c(\boldsymbol{X}_c) = \Phi_{e_c} \circ \cdots \circ \Phi_{s_c}(\boldsymbol{X}_c)$ `// frozen transition`

  Initialize $\boldsymbol{U}_c$

  foreach *token state* $\boldsymbol{x} \in \boldsymbol{X}_c$ **do**

    $\rho_c(\boldsymbol{x}) \leftarrow \text{softmax}(\text{TopK}(\boldsymbol{W}_{\text{gate}}\boldsymbol{x}, k_c))$ `// dynamic routing`

    $g(\boldsymbol{x}) \leftarrow \alpha\boldsymbol{B}\,(\rho(\boldsymbol{x}) \odot \boldsymbol{A}\boldsymbol{x})$ `// token-wise control`

    Append $g(\boldsymbol{x})$ to $\boldsymbol{U}_c$

  $\boldsymbol{X}_{c+1} \leftarrow \boldsymbol{Y}_c + \boldsymbol{U}_c$ `// boundary update`

return $\boldsymbol{X}_{C+1}$

C Theoretical Results

C.1 Proof of Theorem 3.1

Assumption C.1 (Regularity on a reachable region)**.** Fix $\mathcal{G}_c = \{s_c, \dots, e_c\}$, $M_c := |\mathcal{G}_c|$, and let $\boldsymbol{x}_{s_c-1}^0 = \boldsymbol{x}_c$, $\boldsymbol{x}_\ell^0 = \Phi_\ell(\boldsymbol{x}_{\ell-1}^0)$ be the frozen trajectory. Let $\Omega \subset \mathbb{R}^d$ be convex and compact with $\{\boldsymbol{x}_\ell^0\}_{\ell=s_c-1}^{e_c} \subset \text{int}\,\Omega$.

(C1) *(Smooth backbone.)* Each Φ_ℓ is C^1 on Ω: $\|D\Phi_\ell(\boldsymbol{x})\| \le \Lambda$ and $\|D\Phi_\ell(\boldsymbol{x}) - D\Phi_\ell(\boldsymbol{y})\| \le H\|\boldsymbol{x} - \boldsymbol{y}\|$ for all $\boldsymbol{x}, \boldsymbol{y} \in \Omega$.

(C2) *(Bounded controls.)* Each $g_\ell = \boldsymbol{B}_\ell\psi_\ell$ is L_g^ℓ-Lipschitz, and the control strength $\gamma = \max_\ell\{\|g_\ell(\boldsymbol{x}_{\ell-1}^0)\|, L_g^\ell\}$ is finite.

(C3) *(The controlled trajectory stays in Ω.)* $\gamma\,\Gamma_{M_c} \le \text{dist}\big(\{\boldsymbol{x}_\ell^0\}, \partial\Omega\big)$, where

$$\Gamma_j := \sum_{i=0}^{j-1}(\Lambda + \gamma)^i, \qquad \Gamma_0 = 0.$$

We write $\theta := \max\{1, \Lambda + \gamma\}$, so that $\Gamma_j \le j\,\theta^{\,j-1}$.

Proof of Theorem 3.1. **(i) Linear case.** Write $\Phi_\ell(\boldsymbol{x}) = \boldsymbol{W}_\ell\boldsymbol{x}$, $\psi_\ell(\boldsymbol{x}) = \boldsymbol{A}_\ell\boldsymbol{x}$ and $\boldsymbol{R}_\ell := \boldsymbol{B}_\ell\boldsymbol{A}_\ell$, so the fine-grained group map is $\widetilde{F}_c = (\boldsymbol{W}_{e_c} + \boldsymbol{R}_{e_c})\cdots(\boldsymbol{W}_{s_c} + \boldsymbol{R}_{s_c})$ while $F_c = \boldsymbol{W}_{e_c}\cdots\boldsymbol{W}_{s_c}$.

Telescoping $\Delta \boldsymbol{W}_c := \widetilde{F}_c - F_c$ gives

$$\Delta \boldsymbol{W}_c = \sum_{\ell=s_c}^{e_c} \Big(\prod_{i=e_c}^{\ell+1} (\boldsymbol{W}_i + \boldsymbol{R}_i) \Big) \boldsymbol{R}_\ell \Big(\prod_{i=\ell-1}^{s_c} \boldsymbol{W}_i \Big), \tag{12}$$

with empty products equal to $\boldsymbol{I}$. Since $\operatorname{rank}(\boldsymbol{R}_\ell) \le r_\ell$, rank subadditivity together with $\operatorname{rank}(\boldsymbol{ARB}) \le \operatorname{rank}(\boldsymbol{R})$ yields $\operatorname{rank}(\Delta \boldsymbol{W}_c) \le \sum_{\ell=s_c}^{e_c} r_\ell$. Setting $g_{\mathcal{G}_c}(\boldsymbol{x}) := \Delta \boldsymbol{W}_c \boldsymbol{x}$ gives $\boldsymbol{z}_{e_c} = F_c(\boldsymbol{x}_c) + g_{\mathcal{G}_c}(\boldsymbol{x}_c) = \boldsymbol{x}_{c+1}$ with the claimed rank bound.

(ii) Nonlinear case. Let $\boldsymbol{d}_\ell := \boldsymbol{z}_\ell - \boldsymbol{x}_\ell^0$, so $\boldsymbol{d}_{s_c-1} = \boldsymbol{0}$.

Step 1 (distance alignment). We claim $\|\boldsymbol{d}_\ell\| \le \gamma\, \Gamma_{\ell - s_c + 1}$ for every $\ell \in \{s_c - 1, \dots, e_c\}$. The case $\ell = s_c - 1$ holds since $\Gamma_0 = 0$. Assuming the claim at $\ell - 1$, we get $\|\boldsymbol{d}_{\ell-1}\| \le \gamma \Gamma_{M_c}$, so $\boldsymbol{z}_{\ell-1} \in \Omega$ by condition C3, and conditions C1–C2 apply:

$$\begin{aligned} \|\boldsymbol{d}_\ell\| &\le \|\Phi_\ell(\boldsymbol{z}_{\ell-1}) - \Phi_\ell(\boldsymbol{x}_{\ell-1}^0)\| + \|g_\ell(\boldsymbol{z}_{\ell-1}) - g_\ell(\boldsymbol{x}_{\ell-1}^0)\| + \|g_\ell(\boldsymbol{x}_{\ell-1}^0)\| \\ &\le (\Lambda + \gamma)\|\boldsymbol{d}_{\ell-1}\| + \gamma \;\le\; (\Lambda + \gamma)\,\gamma \Gamma_{\ell - s_c} + \gamma \;=\; \gamma\, \Gamma_{\ell - s_c + 1}, \end{aligned} \tag{13}$$

where the first inequality uses the mean value inequality for Φ_ℓ on the convex set Ω and the last uses $\Gamma_{j+1} = 1 + (\Lambda + \gamma)\Gamma_j$.

Step 2 (linearized reference). Let $\delta_{s_c - 1} := \boldsymbol{0}$ and $\delta_\ell := J_\ell \delta_{\ell-1} + g_\ell(\boldsymbol{x}_{\ell-1}^0)$. Unrolling,

$$\delta_{e_c} = \sum_{\ell=s_c}^{e_c} J_{e_c:\ell+1}\, g_\ell(\boldsymbol{x}_{\ell-1}^0) = g_{\mathcal{G}_c}(\boldsymbol{x}_c), \tag{14}$$

so $\boldsymbol{x}_{c+1} = \boldsymbol{x}_{e_c}^0 + \delta_{e_c}$ and $\boldsymbol{z}_{e_c} - \boldsymbol{x}_{c+1} = \boldsymbol{d}_{e_c} - \delta_{e_c}$. Writing $\boldsymbol{B}_\ell = [\boldsymbol{b}_{\ell,1}, \dots, \boldsymbol{b}_{\ell,r_\ell}]$,

$$g_{\mathcal{G}_c}(\boldsymbol{x}_c) = \sum_{\ell=s_c}^{e_c} \sum_{p=1}^{r_\ell} \big[\psi_\ell(\boldsymbol{x}_{\ell-1}^0)\big]_p\, J_{e_c:\ell+1} \boldsymbol{b}_{\ell,p} \;\in\; \mathcal{V}_c := \operatorname{Span}\{ J_{e_c:\ell+1} \boldsymbol{b}_{\ell,p} \;:\; \ell \in \mathcal{G}_c,\; 1 \le p \le r_\ell \}, \tag{15}$$

a span of $\sum_{\ell=s_c}^{e_c} r_\ell$ vectors, whence $\dim \mathcal{V}_c \le \sum_{\ell=s_c}^{e_c} r_\ell$. Unlike the linear case, these directions depend on the frozen trajectory and hence on $\boldsymbol{x}_c$, so $\mathcal{V}_c$ is a per-input subspace rather than a globally fixed low-rank output space.

Step 3 (error recursion). Let $\boldsymbol{w}_\ell := \boldsymbol{d}_\ell - \delta_\ell$ denote the discrepancy between the true deviation and its linearization, so $\boldsymbol{w}_{s_c - 1} = \boldsymbol{0}$. Since $\boldsymbol{z}_{\ell-1} = \boldsymbol{x}_{\ell-1}^0 + \boldsymbol{d}_{\ell-1}$, the two recursions read

$$\boldsymbol{d}_\ell = \Phi_\ell(\boldsymbol{x}_{\ell-1}^0 + \boldsymbol{d}_{\ell-1}) - \Phi_\ell(\boldsymbol{x}_{\ell-1}^0) + g_\ell(\boldsymbol{z}_{\ell-1}), \tag{16}$$

$$\delta_\ell = J_\ell \delta_{\ell-1} + g_\ell(\boldsymbol{x}_{\ell-1}^0). \tag{17}$$

Subtracting them and inserting $\pm J_\ell \boldsymbol{d}_{\ell-1}$ in order to isolate the propagated discrepancy $J_\ell(\boldsymbol{d}_{\ell-1} - \delta_{\ell-1}) = J_\ell \boldsymbol{w}_{\ell-1}$ from the linearization residual, we obtain

$$\boldsymbol{w}_\ell = J_\ell \boldsymbol{w}_{\ell-1} + \underbrace{\big[\Phi_\ell(\boldsymbol{x}_{\ell-1}^0 + \boldsymbol{d}_{\ell-1}) - \Phi_\ell(\boldsymbol{x}_{\ell-1}^0) - J_\ell \boldsymbol{d}_{\ell-1}\big]}_{=:\, \boldsymbol{t}_\ell} + \underbrace{\big[g_\ell(\boldsymbol{z}_{\ell-1}) - g_\ell(\boldsymbol{x}_{\ell-1}^0)\big]}_{=:\, \Delta g_\ell}. \tag{18}$$

The insertion is what makes $\boldsymbol{t}_\ell$ a genuine first-order Taylor remainder of Φ_ℓ at $\boldsymbol{x}_{\ell-1}^0$ with increment $\boldsymbol{d}_{\ell-1}$: by C1 and the integral form of the remainder,

$$\boldsymbol{t}_\ell = \int_0^1 \big[D\Phi_\ell(\boldsymbol{x}_{\ell-1}^0 + t\boldsymbol{d}_{\ell-1}) - J_\ell \big] \boldsymbol{d}_{\ell-1}\, dt, \qquad \|\boldsymbol{t}_\ell\| \le \tfrac{H}{2} \|\boldsymbol{d}_{\ell-1}\|^2, \tag{19}$$

while C2 gives $\|\Delta g_\ell\| \le L_g^\ell \|\boldsymbol{d}_{\ell-1}\| \le \gamma \|\boldsymbol{d}_{\ell-1}\|$. Both terms are therefore second order in the control magnitude, for different reasons: $\boldsymbol{t}_\ell$ through the curvature of the frozen layer, and Δg_ℓ because γ bounds the Lipschitz constant of the control while $\|\boldsymbol{d}_{\ell-1}\| = \mathcal{O}(\gamma)$ by Step 1. Taking norms in Equation (18) and using $\|J_\ell\| \le \Lambda$,

$$\|\boldsymbol{w}_\ell\| \le \Lambda \|\boldsymbol{w}_{\ell-1}\| + \tfrac{H}{2} \|\boldsymbol{d}_{\ell-1}\|^2 + \gamma \|\boldsymbol{d}_{\ell-1}\|. \tag{20}$$

Step 4 (propagation to the boundary). Unrolling Equation (20) from $\boldsymbol{w}_{s_c-1} = \mathbf{0}$ gives

$$\|\boldsymbol{w}_{e_c}\| \;\le\; \sum_{\ell=s_c}^{e_c} \Lambda^{\,e_c-\ell}\Big[\tfrac{H}{2}\|\boldsymbol{d}_{\ell-1}\|^2 + \gamma\|\boldsymbol{d}_{\ell-1}\|\Big], \tag{21}$$

where $\Lambda^{\,e_c-\ell}$ is the amplification applied to the error created at layer ℓ on its way to the group boundary. By Step 1 and the monotonicity of Γ_j, every increment obeys $\|\boldsymbol{d}_{\ell-1}\| \le \gamma\Gamma_{\ell-s_c} \le \gamma\Gamma_{M_c-1}$ for $\ell \in \mathcal{G}_c$, so the bracket admits the ℓ-independent bound

$$\tfrac{H}{2}\|\boldsymbol{d}_{\ell-1}\|^2 + \gamma\|\boldsymbol{d}_{\ell-1}\| \;\le\; \tfrac{H}{2}\gamma^2\Gamma_{M_c-1}^2 + \gamma^2\Gamma_{M_c-1} \;=\; \gamma^2\,\Gamma_{M_c-1}\Big(1 + \tfrac{H}{2}\,\Gamma_{M_c-1}\Big), \tag{22}$$

both terms being quadratic in γ: one factor from the coefficient and one from $\|\boldsymbol{d}_{\ell-1}\|$. Factoring $\gamma^2\Gamma_{M_c-1}$ out and re-indexing the geometric sum with $i = e_c - \ell$,

$$\sum_{\ell=s_c}^{e_c} \Lambda^{\,e_c-\ell} = \sum_{i=0}^{M_c-1} \Lambda^i \;\le\; \sum_{i=0}^{M_c-1} (\Lambda+\gamma)^i = \Gamma_{M_c}, \tag{23}$$

so that, using $\boldsymbol{z}_{e_c} - \boldsymbol{x}_{c+1} = \boldsymbol{w}_{e_c}$ from Step 2,

$$\big\|\boldsymbol{z}_{e_c} - \boldsymbol{x}_{c+1}\big\| \;\le\; \underbrace{\Gamma_{M_c}\,\Gamma_{M_c-1}\Big(1 + \tfrac{H}{2}\;\Gamma_{M_c-1}\Big)}_{=:\,C_c}\;\gamma^2. \tag{24}$$

For $M_c = 1$ we have $\Gamma_{M_c-1} = \Gamma_0 = 0$ and hence $C_c = 0$: the only increment entering the sum is $\boldsymbol{d}_{s_c-1} = \mathbf{0}$ (see Remark C.2).

Step 5 (uniformity in γ). C_c still involves γ through Γ_j, but Γ_j and hence C_c are nondecreasing in γ. Thus for any $\gamma_0 > 0$ and all $\gamma \le \gamma_0$, writing $\theta_0 := \max\{1, \Lambda + \gamma_0\}$ and using $\Gamma_{M_c} \le M_c\theta_0^{M_c-1}$ and $\Gamma_{M_c-1} \le (M_c-1)\theta_0^{M_c-2}$,

$$C_c \;\le\; M_c(M_c-1)\,\theta_0^{2M_c-3}\Big(1 + \tfrac{H}{2}\,(M_c-1)\,\theta_0^{M_c-2}\Big), \tag{25}$$

a quantity depending only on $M_c, \Lambda, H, \gamma_0$ and not on γ or $\boldsymbol{x}_c$. This proves Equation (8) and shows the rate $\mathcal{O}(\gamma^2)$ is uniform on $(0, \gamma_0]$; in particular $\|\boldsymbol{z}_{e_c} - \boldsymbol{x}_{c+1}\| \to 0$ as $\gamma \to 0$. This proves the Theorem 3.1. □

Remark C.2 (Singleton groups: the gap vanishes). For $M_c = 1$ the coarse control is exact, with no smoothness required: $e_c = s_c$, the Jacobian product $J_{e_c:s_c+1}$ is empty and equals $\boldsymbol{I}$, so

$$g_{\mathcal{G}_c}(\boldsymbol{x}_c) = \boldsymbol{B}_{s_c}\psi_{s_c}(\boldsymbol{x}^0_{s_c-1}) = g_{s_c}(\boldsymbol{x}_c), \qquad \boldsymbol{x}_{c+1} = \Phi_{s_c}(\boldsymbol{x}_c) + g_{s_c}(\boldsymbol{x}_c) = \boldsymbol{z}_{s_c} = \boldsymbol{z}_{e_c}.$$

Hence, the single summand in Step 4 carries $\|\boldsymbol{d}_{s_c-1}\| \le \gamma\Gamma_0 = 0$. The reason is structural: the group-entry state coincides with the state at which the fine-grained control is evaluated, so no error source in Equation (18) is activated.

C.2 Proof of Proposition 3.2

Proof. Fix the routed trajectory and corresponding active sets $\{\mathcal{S}_c\}_{c=1}^{C}$. For each feature $q \in [R]$, define its cross-group usage count

$$n_q := \sum_{c=1}^{C} \mathbb{1}[q \in \mathcal{S}_c]. \tag{26}$$

Since $|\mathcal{S}_c| = k_c$,

$$\sum_{q=1}^{R} n_q = \sum_{c=1}^{C} |\mathcal{S}_c| = \sum_{c=1}^{C} k_c = K. \tag{27}$$

A feature used by n_q groups contributes $\binom{n_q}{2}$ pairwise overlaps. Hence,

$$\sum_{m<n} |\mathcal{S}_m \cap \mathcal{S}_n| = \sum_{q=1}^{R} \binom{n_q}{2} = \frac{1}{2}\left(\sum_{q=1}^{R} n_q^2 - K\right). \tag{28}$$

By Cauchy–Schwarz,

$$\sum_{q=1}^{R} n_q^2 \geq \frac{1}{R}\left(\sum_{q=1}^{R} n_q\right)^2 = \frac{K^2}{R}. \tag{29}$$

Substituting into Equation (28) gives

$$\sum_{m<n} |\mathcal{S}_m \cap \mathcal{S}_n| \geq \frac{K}{2}\left(\frac{K}{R} - 1\right), \tag{30}$$

which proves Equation (10).

If $K > R$, the lower bound is strictly positive, so $\mathcal{S}_m \cap \mathcal{S}_n \neq \varnothing$ for at least one pair $m < n$.

Finally, since $\bar{k} = K/C$ and $\binom{C}{2} = C(C-1)/2$,

$$\frac{1}{\binom{C}{2}} \sum_{m<n} \frac{|\mathcal{S}_m \cap \mathcal{S}_n|}{\bar{k}} \geq \frac{\frac{K}{2}\left(\frac{K}{R} - 1\right)}{\binom{C}{2}\frac{K}{C}} = \frac{K/R - 1}{C - 1}, \tag{31}$$

which proves Equation (11). □

D Relation to Existing State-Based Controls

We next relate this routing granularity to existing state-based controls. All state-based controls considered here can be written in a common feature-level form. For a group $\mathcal{G}_c$ with input state $\boldsymbol{x}$,

$$g_{\mathcal{G}_c}(\boldsymbol{x}) = \alpha \sum_{c'=1}^{C} \sum_{q} \omega_{c,c',q}(\boldsymbol{x})\, \boldsymbol{b}_q \boldsymbol{a}_q^\top \boldsymbol{x}, \tag{32}$$

where q indexes rank-one features, c' indexes the control site that *owns* feature q, and $\omega_{c,c',q}(\boldsymbol{x})$ specifies how much of that feature is routed into group c. The three methods differ only in ω.

Parallel Control (Zhang et al., 2025). Parallel Control uses independent controllers at each site:

$$\omega^{\mathrm{PC}}_{c,c',q} = \mathbb{1}[c = c']. \tag{33}$$

Thus, each group accesses only its own features, yielding the *no-sharing* case.

Mixture-of-Control (Nguyen et al., 2026). MoC routes complete controllers across depth. Its router acts on the sequence-level representation $\bar{\boldsymbol{x}} = \frac{1}{m_X}\sum_t \boldsymbol{x}^{(t)}$, so a single decision is shared by all tokens: let $\pi(\boldsymbol{X}) \in \mathbb{R}^C$ denote the resulting distribution over the C controllers, with $\pi_{c'}(\boldsymbol{X})$ the weight assigned to the source controller c', and let β be the mixing hyperparameter. The communication strategy is

$$\omega^{\mathrm{MoC}}_{c,c',q}(\boldsymbol{X}) = \beta\, \mathbb{1}[c = c'] + (1 - \beta)\, \pi_{c'}(\boldsymbol{X}), \tag{34}$$

which is independent of q: every feature inside controller c' receives the same coefficient.

Mixture-of-Features (Ours). MoF draws from a single shared pool, so ownership disappears: every group may use any feature, and the coefficient depends on the feature itself,

$$\omega^{\mathrm{MoF}}_{c,c',q}(\boldsymbol{x}) = \rho_{c,q}(\boldsymbol{x})\,\mathbb{1}[c'=1], \tag{35}$$

where the single pool is indexed by $c'=1$ and ρ_c is the sparse router of Equation (9). The dependence on q distinguishes MoF from MoC: routing is performed per rank-one feature, and per token state rather than per pooled sequence, enabling flexible feature reuse across groups while retaining the efficiency of a compact shared pool.

E Experimental Details

This section provides comprehensive details about our experimental setup, including hyperparameter configurations, dataset specifications, and implementation details for all models across different tasks evaluated in this work.

E.1 Dataset Details

To comprehensively evaluate the effectiveness and generalizability of our proposed method, we conduct experiments across a diverse collection of benchmarks spanning natural language generation, vision, reasoning and time-series forecasting.

Language For language tasks, we focus on Natural Language Generation (NLG), where we utilize eight benchmark datasets: BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), Social IQa (Sap et al., 2019), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2021), ARC-Easy/ Challenge (Clark et al., 2018), and OpenBookQA (Mihaylov et al., 2018). These datasets assess model understanding of complex reasoning, physical commonsense, and social dynamics.

Vision. For vision tasks, we adopt the Visual Task Adaptation Benchmark (VTAB-1K) (Zhai et al., 2019), which consists of 19 image classification tasks with only 1,000 labeled training examples per task. The benchmark covers diverse visual domains and is organized into three categories: Natural, consisting of images from natural scenes and objects; Specialized, containing images from domains such as medicine and remote sensing; and Structured, which emphasizes geometric, spatial, and structural reasoning.

Reasoning. For reasoning tasks, we fine-tune on OpenR1-Math-220k (Hugging Face, 2025), an open reconstruction of mathematical reasoning data generated with DeepSeek-R1 (Guo et al., 2025). We use the default configuration and train on curated `(problem, verified trace)` pairs, masking the loss to the assistant reasoning tokens. For training efficiency, we use a 4,096-token budget and discard longer examples rather than truncating them, yielding approximately 37.6k long-CoT training examples. We evaluate on GSM8K (Cobbe et al., 2021), MATH-500 (Hendrycks et al., 2021), and AMC23 (American Mathematics Competition 2023)[1]. GSM8K contains 1,319 grade-school math word problems requiring multi-step arithmetic reasoning; MATH-500 consists of 500 competition-level problems spanning diverse mathematical domains; and AMC23 contains 40 problems from the 2023 American Mathematics Competitions. Together, these benchmarks span mathematical reasoning from grade-school word problems to competition-level mathematics.

Time Series. For time-series forecasting, we evaluate our method on three widely used benchmarks: ETT, Electricity (ECL), and Weather.

ETT (Electricity Transformer Temperature). We use ETTh1 and ETTm1 from the ETT benchmark (Zhou et al., 2021)[2]. The dataset contains two years of measurements collected from electricity transformers in China. Each observation consists of six power-load features and the oil temperature (`OT`) as the target variable. ETTh_1 is recorded at an hourly resolution, whereas ETTm_1 is recorded every 15 minutes. Following the standard benchmark protocol, the data are chronologically divided into training, validation, and test sets.

[1] The AMC23 dataset is publicly available at `https://maa.org/subject/amc/`

[2] The ETT dataset is publicly available at `https://github.com/zhouhaoyi/ETDataset`.

Electricity (ECL). The Electricity Load Diagrams dataset[3] contains electricity consumption measurements from 321 clients. Following prior work (Li et al., 2019), the data are converted into hourly consumption measurements and used for multivariate time-series forecasting. We follow the standard chronological training, validation, and test partition adopted in prior forecasting studies.

Weather. The Weather dataset[4] contains multivariate climatological measurements collected from U.S. weather stations. The observations include variables such as temperature, humidity, pressure, and wind-related measurements. We follow the standard preprocessing and chronological training, validation, and test split commonly used in long-term time-series forecasting benchmarks.

E.2 Model Architecture and Training Configuration

Natural Language Generation Tasks. We utilized some Large scale models as the backbones, including LLaMA2-7B (Touvron et al., 2023), Mistral-7B-Instruct (Jiang et al., 2023), Gemma3-12B-Instruct (Team et al., 2025), and Qwen2.5-14B-Instruct (Qwen et al., 2025). These pretrained models are also finetuned across our methods as well as the baselines. The detailed hyperparameter configurations for the decoder-only models are illustrated in Table 10.

Vision Tasks. We finetune ViT-Tiny/Base backbones (Pretrained on `ImageNet-21K` and then Finetuned on `ImageNet-1K`) on VTAB datasets, under batch size 32, 100 epochs, learning rate 0.001, learning decay 0.0001 to ensure fair comparison.

Mathematical Reasoning Tasks. We fine-tune two instruction-tuned backbones of different architecture: the dense Qwen2.5-7B-Instruct (Qwen et al., 2025) (L=28 layers, d=3584) and the Mixture-of-Experts Qwen3-30B-A3B-Instruct (Yang et al., 2025) (L=48, d=2048, 128 experts per layer with 8 routed per token; 30.5B total and 3.3B active parameters). On the dense backbone, modules are inserted at `q_proj`, `k_proj`, `v_proj`, `up_proj` and `down_proj`; on the MoE backbone we insert LoRA to the attention projections `q_proj`, `k_proj`, `v_proj` and `o_proj`, since the feed-forward projections are replicated inside all 128 experts of every layer and targeting them by name would instantiate 6,144 adapted projections per model instead of 48, making rank calibration meaningless. Within each backbone, every method receives the same trainable-parameter budget (48.4M, 0.64%, on Qwen2.5-7B-Instruct; 137.6M, 0.45%, on Qwen3-30B-A3B), so differences in accuracy reflect where capacity is placed rather than how much of it there is. All runs share a single recipe with the backbone frozen in bfloat16: one epoch, batch size 16, AdamW at learning rate 3×10^{-4} with linear decay after 100 warmup steps, weight decay 0.01, maximum sequence length 4,096. Qwen2.5-7B-Instruct trains on a single H100 80GB GPU, while Qwen3-30B-A3B requires two GPUs H100 80GB under model parallelism, as its bfloat16 weights exceed a single card; checkpoints are decoded greedily within the same 4,096-token budget used in training.

Time-Series Tasks. We fine-tune GPT-2 (12 layers, d=768) and GPT-2-Large (36 layers, d=1280) (Radford et al., 2019) for long-term forecasting, keeping the pretrained self-attention and feed-forward blocks frozen. Each input window is instance-normalised, cut into patches of length 16 with stride 8 (16 for ETTm1), and linearly embedded; following the channel-independent convention, one sample is a single (variate, window) pair. The look-back is 512 steps (336 for ETTh1) and we train a separate model for each horizon $H \in \{96, 192, 336, 720\}$. Training minimises MSE for at most 10 epochs with early stopping (patience 3) on validation MSE, which also selects the reported checkpoint. We use Adam at 10^{-4} for GPT-2 and AdamW at 5×10^{-5} for GPT-2-Large, a cosine schedule ($T_{\max}$=20, $\eta_{\min}$=10^{-8}) stepped per epoch, and dropout 0.3.

For ETTh1, ETTm1 and Weather every epoch covers the full training split. Electricity has 321 variates and thus $\sim$5.7M channel-independent samples per epoch, while consecutive windows overlap by $T-1$ steps; we therefore cap each epoch at 700 batches (716,800 samples at batch size 1024 for GPT-2; 89,600 at batch size 128 for GPT-2-Large), drawn uniformly without replacement and resampled every epoch so that the union over epochs still covers the split. Electricity validation

[3]The ECL dataset is publicly available at `https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014`.

[4]The Weather dataset is publicly available from the NOAA Local Climatological Data repository at `https://www.ncei.noaa.gov/data/local-climatological-data/`.

Table 9: Time-series training configurations. For ETTh1, ETTm1, and Weather, each epoch covers the full training split. For Electricity, each epoch is capped at 700 batches due to the substantially larger number of channel-independent samples.

Dataset	Sampling	T	#Vars	Patch/Stride	Batch Size (Base/Large)	Precision	Samples/Epoch
Weather	10 min	512	21	16/8	64 / 8	fp32	Full split
ETTh1	1 h	336	7	16/8	256 / 32	fp32	Full split
ETTm1	15 min	512	7	16/16	256 / 32	fp32	Full split
Electricity	1 h	512	321	16/8	1024 / 128	bf16	700 batches

Table 10: Hyperparameter settings and Mixture-of-Control configuration for LLaMA2-7B, Mistralv3-7B-Instruct, Gemma3-12B-Instruct, and Qwen2.5-14B-Instruct. Settings are chosen to balance performance and training efficiency.

Parameter	LLaMA2-7B	Mistralv3-7B	Gemma3-12B	Qwen2.5-14B
Learning rate	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}
Batch size	16	16	12	12
Gradient accumulation steps	2	2	2	2
Max sequence length	256	256	256	256
Training epochs	3	3	3	3
Weight decay	0.01	0.01	0.01	0.01

uses one fixed seed-derived subset of 50,000 samples, keeping the early-stopping signal comparable across epochs; not subsampled in the test set. Table 9 gives the per-dataset configuration.

SURF Configurations. Across all experiments, we consider alpha scaling $\alpha = 1$, the grouping threshold over $\tau \in \{0.80, 0.81, \ldots, 0.99\}$, consider both the **Fixed** and **Adaptive** allocation schemes, and select the routing budget from Top-$k \in \{8, 16, 32, 64, 128\}$. For the shared feature-pool size R, we construct the search space relative to the parameter budget of the corresponding Control baseline. Let N denote the number of control sites, (e.g., $N = 2L$ for Control/ MoC), d the hidden dimension, and r the rank of each Control module. The parameter budgets can be decomposed as

$$P_{\text{Control}} = \underbrace{Ndr}_{\text{down projections}} + \underbrace{Nrd}_{\text{up projections}} = 2dNr,$$

$$P_{\text{SURF}} = \underbrace{dR}_{\text{shared down features}} + \underbrace{Rd}_{\text{shared up features}} + \underbrace{dR}_{\text{shared routing weights}} = 3dR.$$

Matching the SURF parameter budget to that of the original Control baseline therefore gives

$$P_{\text{SURF}} = P_{\text{Control}} \Longrightarrow 3dR = 2dNr$$
$$\Longrightarrow \boxed{R = \frac{2Nr}{3}}.$$

We use this equal-budget pool size as a reference and additionally consider smaller pool sizes to study the trade-off between parameter efficiency and adaptation performance.

Inference Throughput. We measure the inference throughput of different LLMs across batch sizes, as shown in Figure 3. Given 100 random samples, we run batched inference and compute throughput as the number of generated tokens divided by the inference time for each batch. We then report the average throughput across all batches for each model.

Compute Resources. The compute resource used in our experiments consists of 8 NVIDIA H100 GPUs, each with 80GB of memory.

Table 11: Single-request inference latency on Mistralv3-7B with batch size 1, KV caching, and 128 generated tokens. We report autoregressive decoding latency (ms/token) and prefill latency (ms) for prompt lengths of 512 and 1024. Lower is better.

	Prompt Length 512			Prompt Length 1024		
Method	Decode (ms/tok)	vs. LoRA	Prefill (ms)	Decode (ms/tok)	vs. LoRA	Prefill (ms)
LoRA (Merged)	$\mathbf{13.045 \pm 0.005}$	$\mathbf{1.00\times}$	$\mathbf{34.6}$	$\mathbf{13.210 \pm 0.002}$	$\mathbf{1.00\times}$	$\mathbf{57.2}$
MoDE	18.289 ± 0.003	$1.40\times$	43.1	18.418 ± 0.006	$1.39\times$	69.8
MoLEx	51.189 ± 0.481	$3.92\times$	86.2	54.399 ± 1.008	$4.12\times$	130.9
Control	14.377 ± 0.005	$1.10\times$	37.3	14.529 ± 0.003	$1.10\times$	60.7
MoC	36.878 ± 0.114	$2.83\times$	56.6	36.224 ± 0.386	$2.74\times$	78.9
SURF	$\mathbf{13.678 \pm 0.003}$	$\mathbf{1.05\times}$	$\mathbf{36.7}$	$\mathbf{13.821 \pm 0.003}$	$\mathbf{1.05\times}$	$\mathbf{60.9}$

E.3 Additional Experiments

Vision Tasks. We report detailed results for all methods across the 19 VTAB datasets and the two pretrained models, together with radar-chart visualizations for a more comprehensive comparison. See Table 16 and Figure 6 for details.

Single-Token Autoregressive Decoding Latency. Beyond batched throughput, we evaluate single-request autoregressive decoding with KV caching, where each decoding step processes one new token ($m_X = 1$). As shown in Table 11, SURF incurs only a $1.05\times$ latency overhead relative to merged LoRA, requiring 13.68 and 13.82 ms/token for prompt lengths of 512 and 1024, respectively. This is lower than Control ($1.10\times$), MoDE (1.39–$1.40\times$), MoC (2.74–$2.83\times$), and MoLEx (3.92–$4.12\times$). Thus, dynamic Top-k feature routing and groupwise projections introduce limited additional latency even in the batch-1 KV-cached regime. Decode latency also remains nearly unchanged as the prompt length doubles, consistent with effective KV caching. Together with the batched-throughput results, these results show that SURF improves throughput while remaining close to the decoding latency of a statically merged adapter.

Time-Series Tasks. We report detailed results for four representative methods across the 4 time-series datasets and the two pretrained models, across different output prediction lengths. See Table 12 and Table 13 for details.

Feature Communication. We analyze feature reuse across different pool sizes R, Top-k budgets, and fixed/adaptive routing schemes. For each setting, we evaluate 256 randomly sampled prompts from each of the eight commonsense reasoning datasets under $\tau = 0.96$, yielding 28 groups. Since feature selection is token-dependent, we summarize routing by how frequently each feature is selected. Let $C_{c,r}$ be the number of valid tokens for which group c selects feature r, and let N_c be the total number of valid tokens processed by that group. We define the empirical selection frequency $\nu_{c,r} = C_{c,r}/N_c$, i.e., the fraction of tokens at group c that select feature r. Since each token selects k_c features, $\sum_{r=1}^{R} \nu_{c,r} = k_c$.

We report three quantities. **Budget Reuse** is $1 - \sum_r \max_c \nu_{c,r} / \sum_c k_c$ and measures how much of the aggregate routing budget is reused across groups rather than assigned to group-specific features. **Shared Features** is the share of the aggregate selection frequency that is duplicated across groups, $\sum_r \nu_r^{(2)} / \sum_r \max_c \nu_{c,r}$, where $\nu_r^{(2)}$ denotes the second-largest selection frequency at feature r; it is 0 when every feature serves a single group and approaches 1 when all groups route identically. **Average Pairwise Jaccard** averages the frequency-weighted overlap $\sum_r \min(\nu_{i,r}, \nu_{j,r}) / \sum_r \max(\nu_{i,r}, \nu_{j,r})$ over all $\binom{28}{2} = 378$ group pairs. All three reduce to their set-based counterparts when ν is binary.

Because each feature can contribute at most one unit of selection frequency to a group, a finite pool necessarily induces reuse when $\sum_c k_c > R$. In this regime Budget Reuse is bounded below by $1 - R/\sum_c k_c$ and above by $1 - \max_c k_c / \sum_c k_c$; the lower bound is attained only by maximally disjoint routing and is therefore a floor, not a target. See Figures 7 to 15 for the details.

Table 12: Full long-term forecasting results using GPT2. We use prediction length $H \in \{96, 192, 336, 720\}$. Lower MSE and MAE indicate better performance. **Black**: best, **Red**: second best.

Methods		LoRA		Control		MoC		SURF	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	0.152	0.205	0.150	0.201	0.154	0.204	0.148	0.201
	192	0.195	0.242	0.200	0.249	0.206	0.254	0.193	0.243
	336	0.251	0.286	0.248	0.287	0.272	0.299	0.246	0.281
	720	0.326	0.338	0.324	0.339	0.344	0.347	0.315	0.331
	Avg	**0.231**	**0.268**	**0.231**	0.269	0.244	0.276	**0.226**	**0.264**
ETTh1	96	0.388	0.406	0.378	0.399	0.386	0.403	0.360	0.386
	192	0.436	0.432	0.424	0.423	0.419	0.421	0.408	0.411
	336	0.457	0.446	0.454	0.440	0.452	0.441	0.440	0.438
	720	0.492	0.480	0.510	0.485	0.479	0.472	0.460	0.463
	Avg	0.443	0.441	0.441	0.437	**0.434**	**0.434**	**0.417**	**0.425**
ETTm1	96	0.303	0.353	0.305	0.352	0.293	0.350	0.290	0.345
	192	0.333	0.372	0.338	0.376	0.337	0.376	0.324	0.365
	336	0.368	0.392	0.370	0.396	0.376	0.395	0.358	0.382
	720	0.420	0.429	0.414	0.426	0.426	0.420	0.406	0.405
	Avg	**0.356**	0.386	0.357	0.387	0.358	**0.385**	**0.345**	**0.374**
ECL	96	0.135	0.231	0.133	0.228	0.135	0.230	0.130	0.225
	192	0.151	0.246	0.151	0.246	0.151	0.246	0.150	0.244
	336	0.166	0.261	0.165	0.260	0.167	0.262	0.164	0.258
	720	0.206	0.295	0.204	0.292	0.206	0.295	0.204	0.294
	Avg	0.165	0.258	**0.163**	**0.257**	0.165	0.258	**0.162**	**0.255**
Average		**0.299**	0.338	0.298	**0.337**	0.300	0.338	**0.287**	**0.329**

Table 13: Full long-term forecasting results using GPT2-Large. We use prediction length $H \in \{96, 192, 336, 720\}$. Lower MSE and MAE indicate better performance. **Black**: best, **Red**: second best.

Methods		LoRA		Control		MoC		SURF	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	0.158	0.208	0.161	0.211	0.155	0.207	0.150	0.199
	192	0.228	0.263	0.208	0.265	0.219	0.262	0.199	0.247
	336	0.314	0.324	0.290	0.307	0.275	0.301	0.251	0.287
	720	0.398	0.380	0.384	0.374	0.375	0.380	0.319	0.339
	Avg	0.275	0.294	0.261	0.289	**0.256**	**0.288**	**0.230**	**0.268**
ETTh1	96	0.387	0.406	0.400	0.418	0.409	0.419	0.370	0.390
	192	0.436	0.435	0.460	0.447	0.439	0.439	0.425	0.423
	336	0.519	0.495	0.445	0.445	0.459	0.454	0.436	0.434
	720	0.485	0.476	0.477	0.468	0.570	0.530	0.475	0.468
	Avg	0.457	0.453	**0.446**	**0.444**	0.469	0.461	**0.427**	**0.429**
ETTm1	96	0.316	0.364	0.320	0.365	0.322	0.362	0.302	0.350
	192	0.359	0.386	0.352	0.387	0.362	0.390	0.343	0.381
	336	0.385	0.404	0.405	0.414	0.378	0.402	0.377	0.402
	720	0.455	0.449	0.442	0.439	0.440	0.443	0.418	0.427
	Avg	0.379	0.401	0.380	0.401	**0.375**	**0.399**	**0.360**	**0.390**
ECL	96	0.137	0.235	0.136	0.234	0.141	0.239	0.130	0.229
	192	0.154	0.250	0.152	0.246	0.155	0.252	0.145	0.244
	336	0.168	0.264	0.167	0.264	0.168	0.264	0.164	0.257
	720	0.211	0.301	0.206	0.295	0.212	0.302	0.201	0.289
	Avg	0.167	0.262	**0.165**	**0.260**	0.169	0.264	**0.160**	**0.255**
Average		0.319	0.353	**0.313**	**0.349**	0.317	0.353	**0.294**	**0.335**

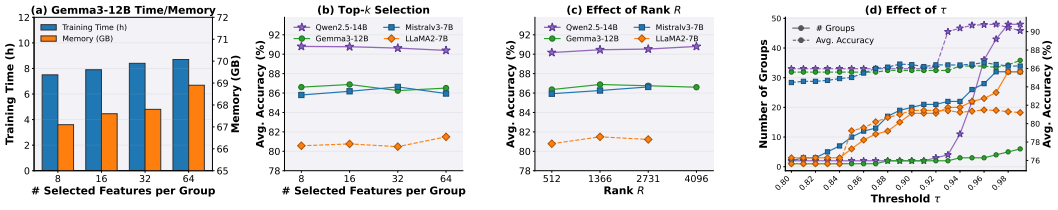

Figure 4: SURF ablations: (a) training time and GPU memory versus features per group on Gemma3-12B-Instruct; (b–d) performance across LLM backbones under varying (b) feature budget, (c) pool size R, and (d) cosine threshold τ. Colors in (d) follow (b–c).

Table 14: Ablation of SURF components under fixed and adaptive settings.

Model	SURF	Fixed (↑)	Adaptive (↑)
Mistralv3-7B	**Full**	**86.6**	**86.2**
	w/o grouping	86.1	–
	w/o MoF	86.2	85.2
Gemma3-12B	**Full**	**86.7**	**86.9**
	w/o grouping	85.1	–
	w/o MoF	84.4	85.6

Table 15: Ablation of SURF across generalized control families on VTAB benchmarks.

Model	Activation	Natural (↑)	Specialized (↑)	Structured (↑)	Avg (↑)
ViT-Tiny	Identity	69.4	**85.5**	**49.3**	**64.3**
	SiLU	**69.6**	84.3	47.5	63.4
	Tanh	68.6	83.0	45.7	62.0
ViT-Base	Identity	**80.3**	88.5	**54.1**	**71.0**
	SiLU	78.1	**89.1**	52.4	69.6
	Tanh	77.9	87.4	50.7	68.4

F Ablation Studies

G Discussion

Beyond layerwise control. Layerwise discretization removes the immediate intra-layer overhead of state-based adaptation, but a depth-L model still requires up to L explicit state-dependent interventions. In Section 3, we address this issue through an inexpensive offline similarity-guided grouping procedure, which provides an effective model-dependent strategy for reducing the number of control evaluations. However, such an offline, preprocessing scheme may not be equally suitable for every architecture or task. This raises a broader question: "can the locations of state interventions be identified directly through optimization, rather than prescribed by an offline grouping rule?"

This perspective is closely related to the classical *maximum hands-off control* problem (Nagahara et al., 2015), which seeks control trajectories with minimal support while satisfying a prescribed control objective. Motivated by this sparsity-promoting principle, we consider a simple two-stage formulation. After training a layerwise controller, we introduce a learnable scalar gate u_ℓ, initialized as 1 for each layer:

$$\boldsymbol{z}_\ell = \Phi_\ell(\boldsymbol{z}_{\ell-1}) + u_\ell g_\ell(\boldsymbol{z}_{\ell-1}), \tag{36}$$

and optimize

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \lambda \sum_{\ell=1}^{L} |u_\ell|. \tag{37}$$

The ℓ_1 penalty encourages minimizing intervention without compromising the performance, allowing the optimization process itself to determine which control sites remain important, without assuming a priori that a particular layer is redundant.

We finetune over 3 epochs, learning rate 0.01, batch size/ micro batch size 16/ 8 for LLaMA2/ Mistralv3-7B-Instruct, 12/ 6 for Gemma3-12B-Instruct/ Qwen2.5-14B-Instruct. As shown in Figure 5, the learned intervention patterns vary substantially across backbones, suggesting that the need for control is inherently model-dependent. For some models and regularization strengths, only a subset of layers remains strongly active, while others retain more distributed control across depth. These observations suggest a more general direction in which intervention locations are learned jointly with adaptation, potentially replacing fixed grouping with data-driven sparse control placement. Such a formulation could further reduce inference overhead and provide a natural interface for implementation-level optimizations that exploit sparse intervention patterns. We leave the development of a fully joint and dynamically learned intervention scheme to future work.

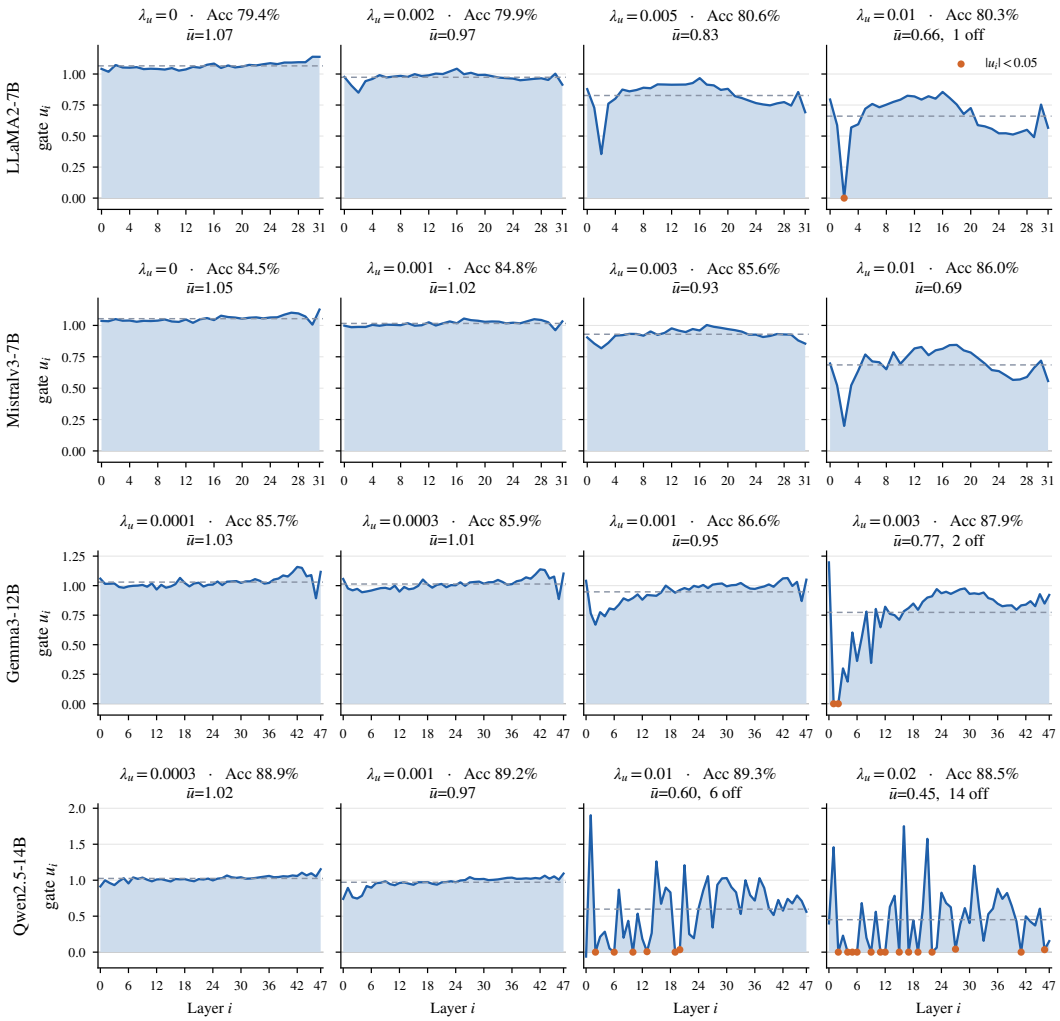

Figure 5: **Exploring learned sparse intervention through two-stage optimization.** Starting from a trained Layerwise Control model, we introduce learnable layer-wise gates u_ℓ and optimize them with an ℓ_1 penalty controlled by λ. We report the resulting gate behavior together with the average accuracy over eight commonsense reasoning datasets across four model backbones. The heterogeneous sparsity patterns suggest that intervention importance is model-dependent and motivate future work on jointly learning where state-based controls should be applied. Such sparse intervention schemes may further narrow the inference-efficiency gap with mergeable weight-based adaptation and enable implementation-level optimizations that skip inactive control sites.

Table 16: VTAB-1K results evaluated by Top-1 accuracy. All methods fine-tune the comparable set of LoRA parameters on ViT-Tiny/Base models pre-trained on ImageNet-21K and subsequently fine-tuned on ImageNet-1K.

	Natural							Specialized				Structured								
Method	Cifar100	Caltech101	DTD	Flower102	Pets	SVHN	Sun397	Camelyon16	EuroSAT	Resisc45	Retinopathy	Clevr-Count	Clevr-Dist	DMLab	KITTI	dSpr-Loc	dSpr-Ori	sNORB-Azi	sNORB-Ele	**Mean**
ViT-Tiny																				
LoRA	31.5	83.5	55.0	95.0	88.5	83.5	23.5	88.0	93.5	78.0	78.0	74.5	61.0	43.0	76.5	39.0	30.5	14.0	**39.5**	61.9
DoRA	32.0↑0.5	80.5↓3.0	**57.5**↑2.5	94.5↓0.5	87.5↓1.0	**87.5**↑4.0	24.5↑1.0	88.0↑0.0	94.5↑1.0	77.0↓1.0	**81.0**↑3.0	75.5↑1.0	56.5↓4.5	**45.0**↑2.0	74.5↓2.0	35.5↓3.5	31.5↑1.0	15.0↑1.0	38.0↓1.5	61.9↑0.0
MoDE	35.0↑3.5	83.5↑0.0	53.0↓2.0	95.0↑0.0	88.0↓0.5	86.0↑2.5	28.0↑4.5	86.0↓2.0	90.5↓3.0	**80.5**↑2.5	76.5↓1.5	77.0↑2.5	58.5↓2.5	42.5↓0.5	80.0↑3.5	37.0↓2.0	27.0↓3.5	16.0↑2.0	**41.5**↑2.0	62.2↑0.3
MoLEx	34.0↑2.5	88.0↑4.5	56.0↑1.0	90.5↓4.5	86.0↓2.5	85.5↑2.0	23.0↓0.5	**89.0**↑1.0	**96.0**↑2.5	**80.5**↑2.5	76.5↓1.5	70.5↓4.0	57.5↓3.5	**45.0**↑2.0	80.5↑4.0	36.5↓2.5	29.0↓1.5	15.0↑1.0	36.0↓3.5	61.8↓0.1
Control	31.0↓0.5	84.5↑1.0	55.5↑0.5	94.5↓0.5	85.5↓3.0	87.0↑3.5	25.0↑1.5	87.5↓0.5	**96.0**↑2.5	78.5↑0.5	76.5↓1.5	69.5↓5.0	58.0↓3.0	40.0↓3.0	76.0↓0.5	37.5↓1.5	31.5↑1.0	14.5↑0.5	39.5↑0.0	61.5↓0.4
MoC	**38.0**↑6.5	88.0↑4.5	55.0↑0.0	**97.0**↑2.0	87.5↓1.0	87.0↑3.5	27.0↑3.5	85.5↓2.5	92.0↓1.5	77.5↓0.5	75.5↓2.5	76.5↑2.0	56.5↓4.5	40.5↓2.5	78.0↑1.5	38.5↓0.5	28.0↓2.5	13.0↓1.0	37.0↓2.5	62.0↑0.1
SURF	**38.0**↑6.5	**88.5**↑5.0	**57.5**↑2.5	**97.0**↑2.0	**89.5**↑1.0	85.5↑2.0	**29.5**↑6.0	**89.0**↑1.0	94.5↑1.0	79.5↑1.5	79.0↑1.0	**77.5**↑3.0	**62.5**↑1.5	43.5↑0.5	**82.5**↑6.0	**41.0**↑2.0	**32.0**↑1.5	**18.0**↑4.0	37.5↓2.0	**64.3**↑2.4
ViT-Base																				
LoRA	66.0	96.0	62.0	**99.5**	**94.0**	89.0	49.0	89.5	96.0	85.0	77.5	**86.5**	61.5	47.0	82.5	**43.0**	35.0	18.0	42.0	69.4
DoRA	64.0↓2.0	95.5↓0.5	62.5↑0.5	**99.5**↑0.0	93.0↓1.0	90.0↑1.0	48.5↓0.5	87.5↓2.0	96.0↑0.0	82.5↓2.5	79.5↑2.0	84.0↓2.5	60.5↓1.0	46.5↓0.5	84.5↑2.0	38.5↓4.5	34.0↓1.0	19.5↑1.5	43.0↑1.0	68.9↓0.5
MoDE	66.0↑0.0	**97.0**↑1.0	62.0↑0.0	99.0↓0.5	**94.0**↑0.0	89.0↑0.0	48.5↓0.5	89.5↑0.0	97.0↑1.0	85.0↑0.0	77.5↑0.0	83.0↓3.5	60.0↓1.5	44.5↓2.5	84.5↑2.0	37.5↓5.5	34.0↓1.0	16.5↓1.5	37.5↓4.5	68.5↓0.9
MoLEx	65.5↓0.5	**97.0**↑1.0	62.5↑0.5	**99.5**↑0.0	93.0↓1.0	87.5↓1.5	48.5↓0.5	90.0↑0.5	96.5↑0.5	85.0↑0.0	78.5↑1.0	81.5↓5.0	63.0↑1.5	42.5↓4.5	81.0↓1.5	39.0↓4.0	37.5↑2.5	16.0↓2.0	39.0↓3.0	68.6↓0.8
Control	63.0↓3.0	96.0↑0.0	62.0↑0.0	**99.5**↑0.0	92.5↓1.5	**91.5**↑2.5	47.0↓2.0	90.0↑0.5	97.0↑1.0	85.5↑0.5	**80.0**↑2.5	78.0↓8.5	64.5↑3.0	**49.0**↑2.0	82.5↑0.0	42.0↓1.0	36.0↑1.0	18.5↑0.5	38.5↓3.5	69.1↓0.3
MoC	**67.5**↑1.5	96.0↑0.0	**64.0**↑2.0	**99.5**↑0.0	93.0↓1.0	91.0↑2.0	**50.5**↑1.5	88.5↓1.0	97.0↑1.0	85.0↑0.0	78.5↑1.0	84.5↓2.0	65.5↑4.0	47.5↑0.5	83.0↑0.5	40.0↓3.0	36.0↑1.0	16.5↓1.5	38.0↓4.0	69.6↑0.2
SURF	67.0↑1.0	**97.0**↑1.0	62.5↑0.5	**99.5**↑0.0	**94.0**↑0.0	**91.5**↑2.5	**50.5**↑1.5	**90.5**↑1.0	**98.0**↑2.0	**86.0**↑1.0	79.5↑2.0	83.5↓3.0	**66.0**↑4.5	48.5↑1.5	**85.0**↑2.5	**43.0**↑0.0	**40.0**↑5.0	**22.5**↑4.5	**44.5**↑2.5	**71.0**↑1.6

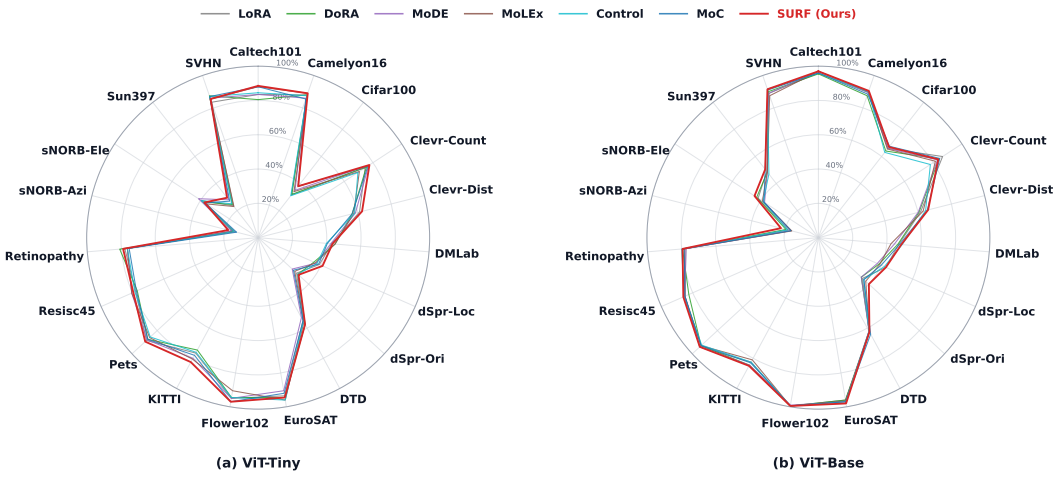

Figure 6: Benchmark comparison of LoRA, DoRA, MoDE, MoLEx, Control, MoC, and SURF on VTAB across 19 datasets, shown separately for ViT-Tiny (left) and ViT-Base (right). Values indicate classification accuracy, with the outer ring corresponding to 100%.

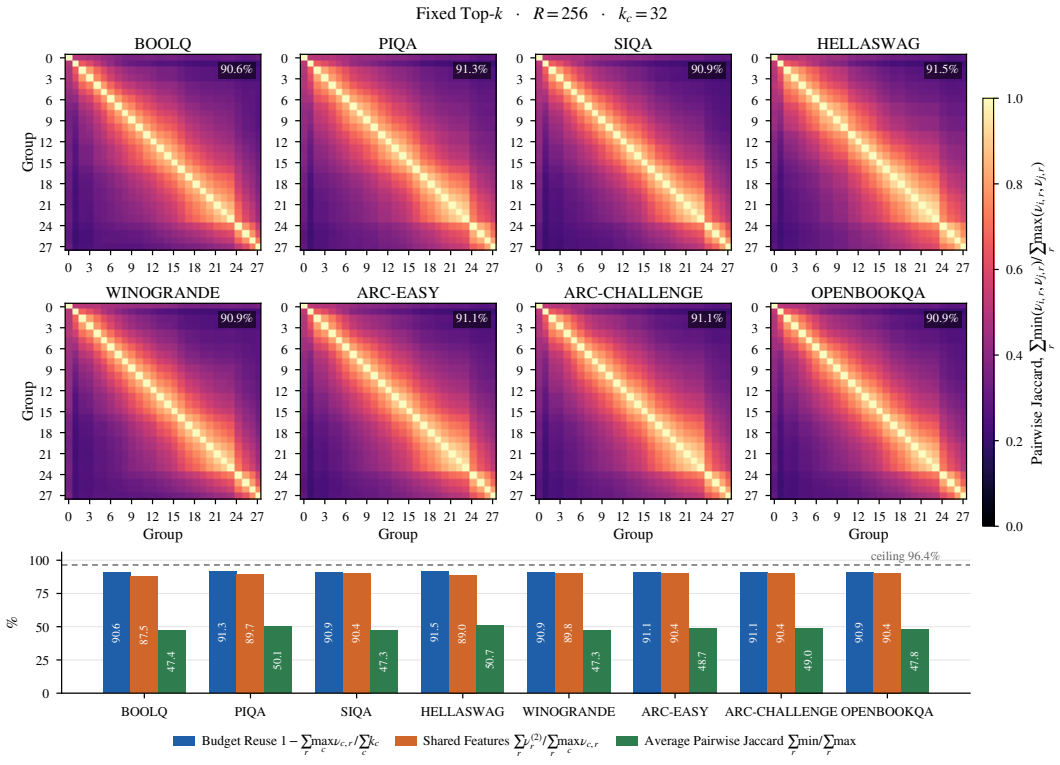

Figure 7: SURF Feature Communication: Fixed Scheme, $R = 256$, $\tau = 0.96$, Top-$k = 32$.

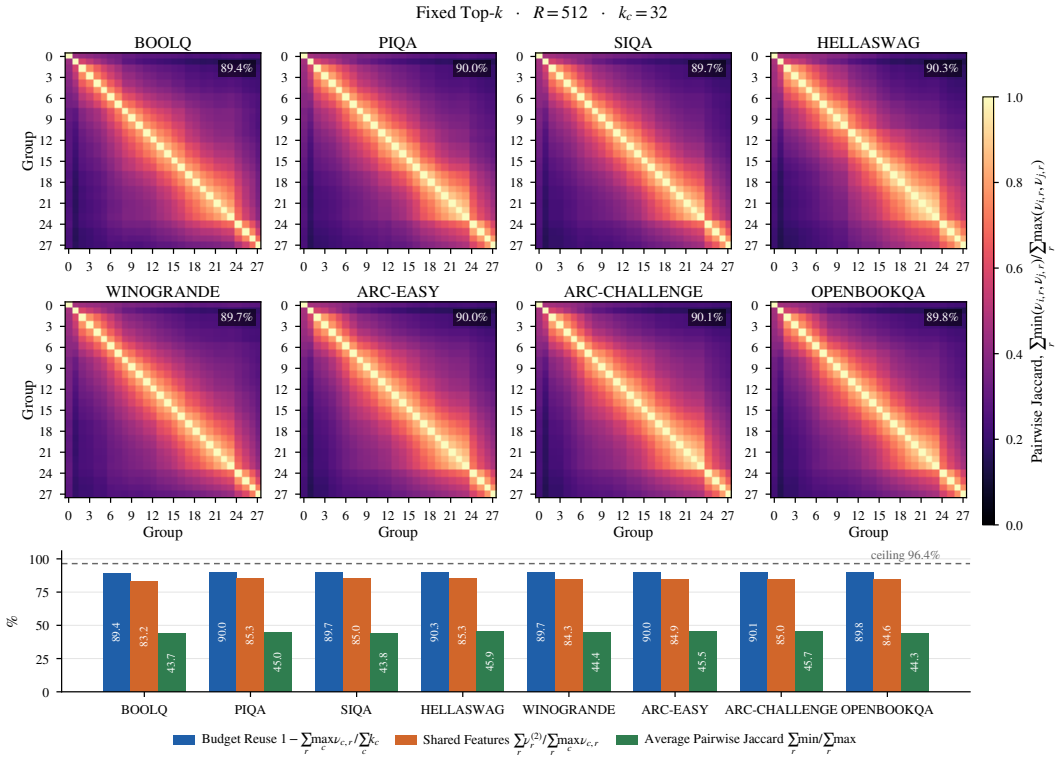

Figure 8: SURF Feature Communication: Fixed Scheme, $R = 512$, $\tau = 0.96$, Top-$k = 32$.

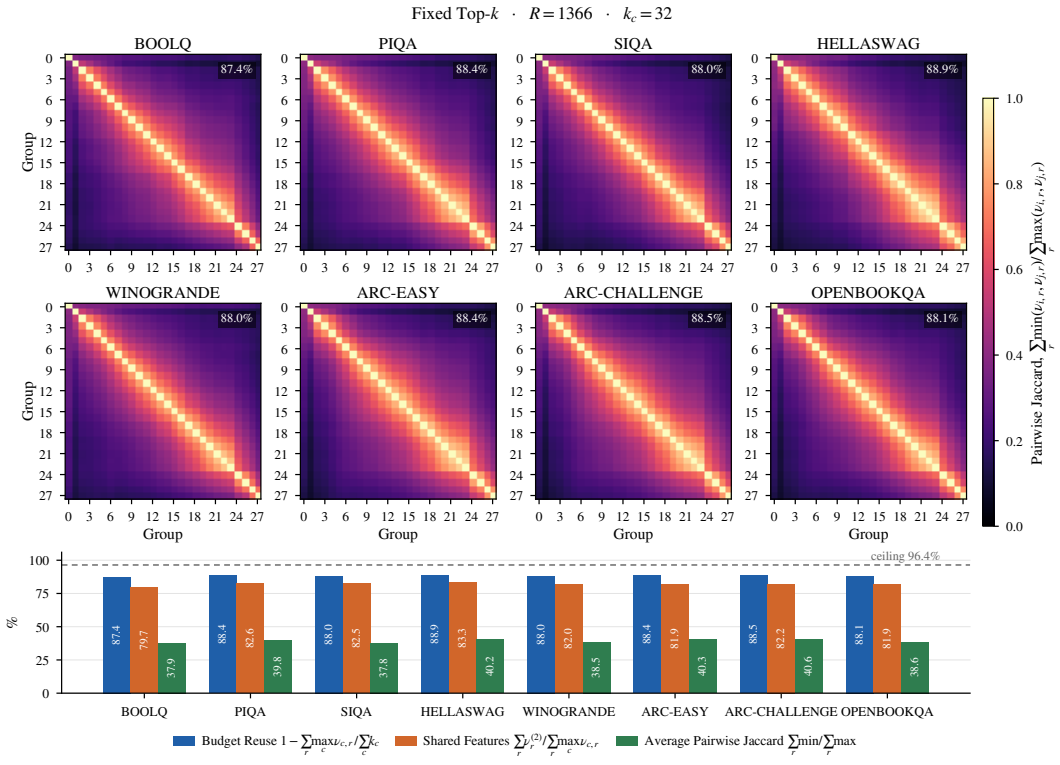

Figure 9: SURF Feature Communication: Fixed Scheme, $R = 1366$, $\tau = 0.96$, Top-$k = 32$.

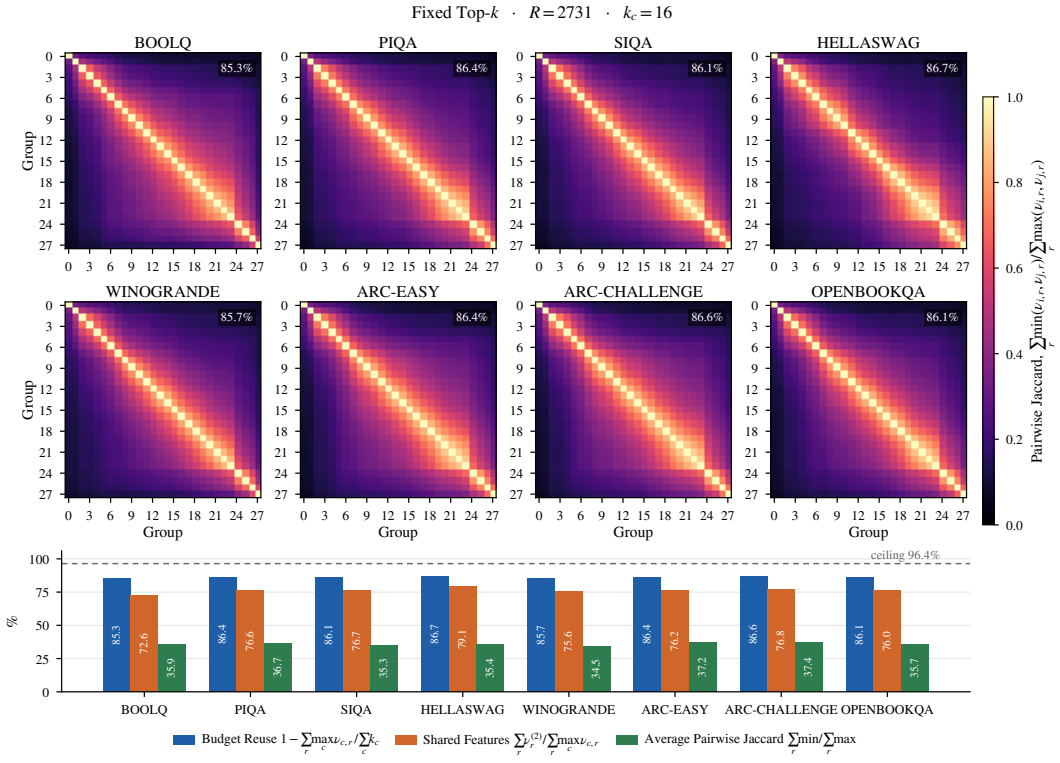

Figure 10: SURF Feature Communication: Fixed Scheme, $R = 2731$, $\tau = 0.96$, Top-$k = 16$.

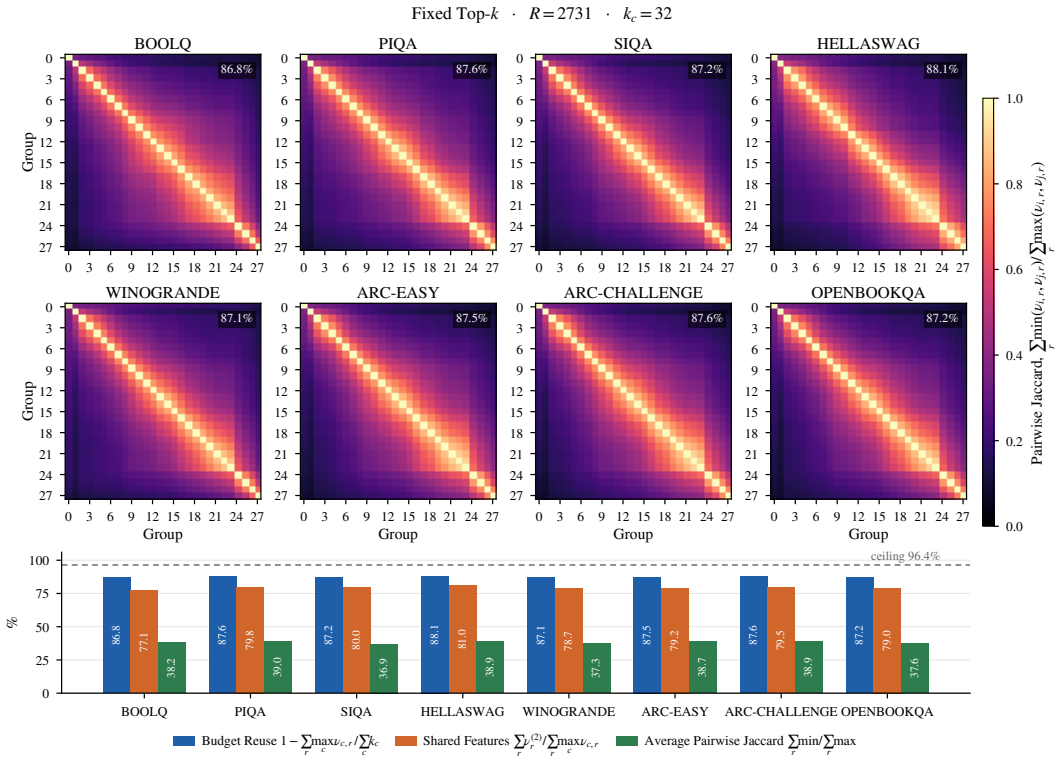

Figure 11: SURF Feature Communication: Fixed Scheme, $R = 2731$, $\tau = 0.96$, Top-$k = 32$.

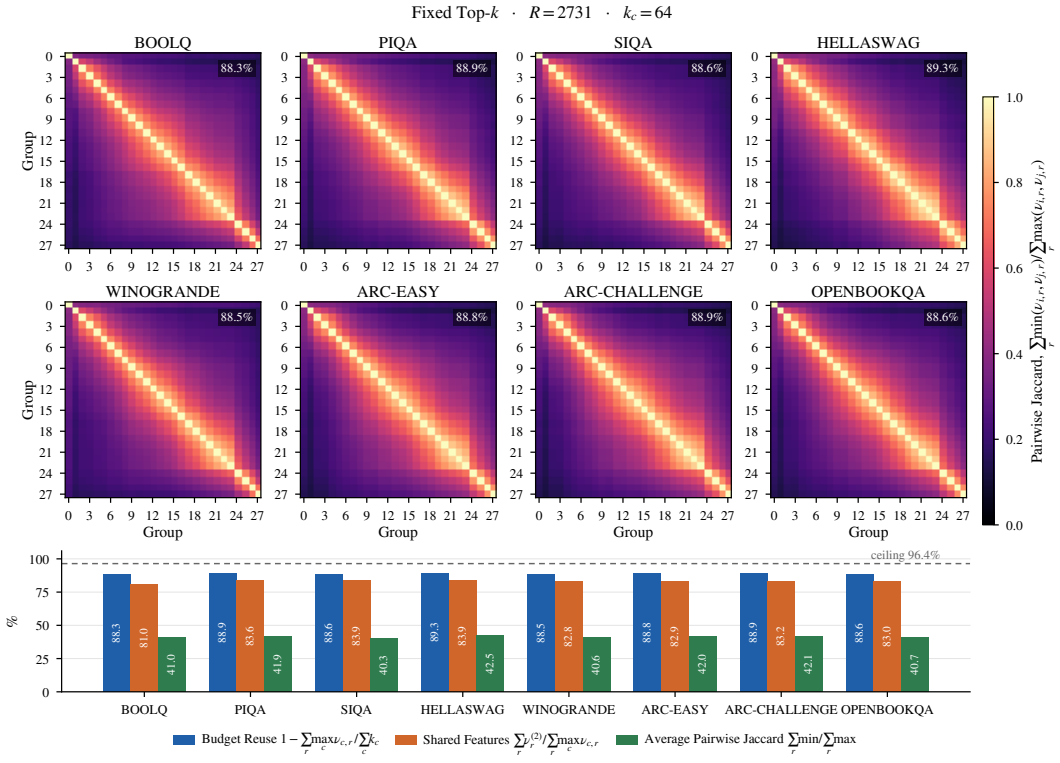

Figure 12: SURF Feature Communication: Fixed Scheme, $R = 2731$, $\tau = 0.96$, Top-$k = 64$.

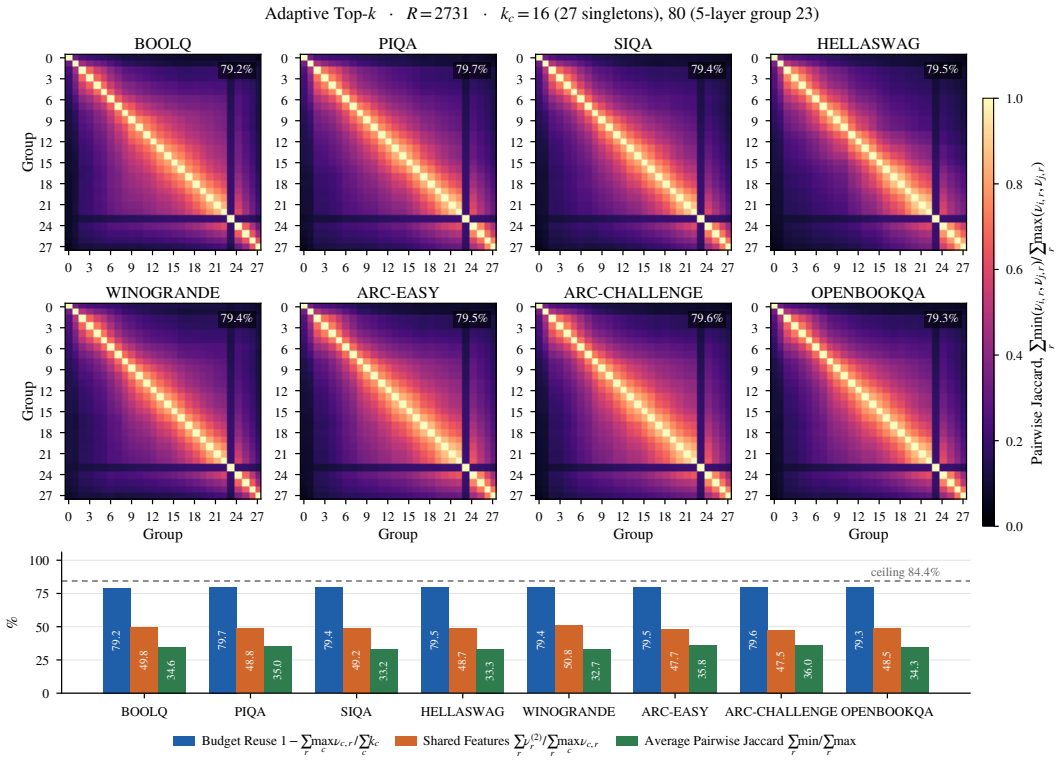

Figure 13: SURF Feature Communication: Adaptive Scheme, $R = 2731$, $\tau = 0.96$, Top-$k = 16$.

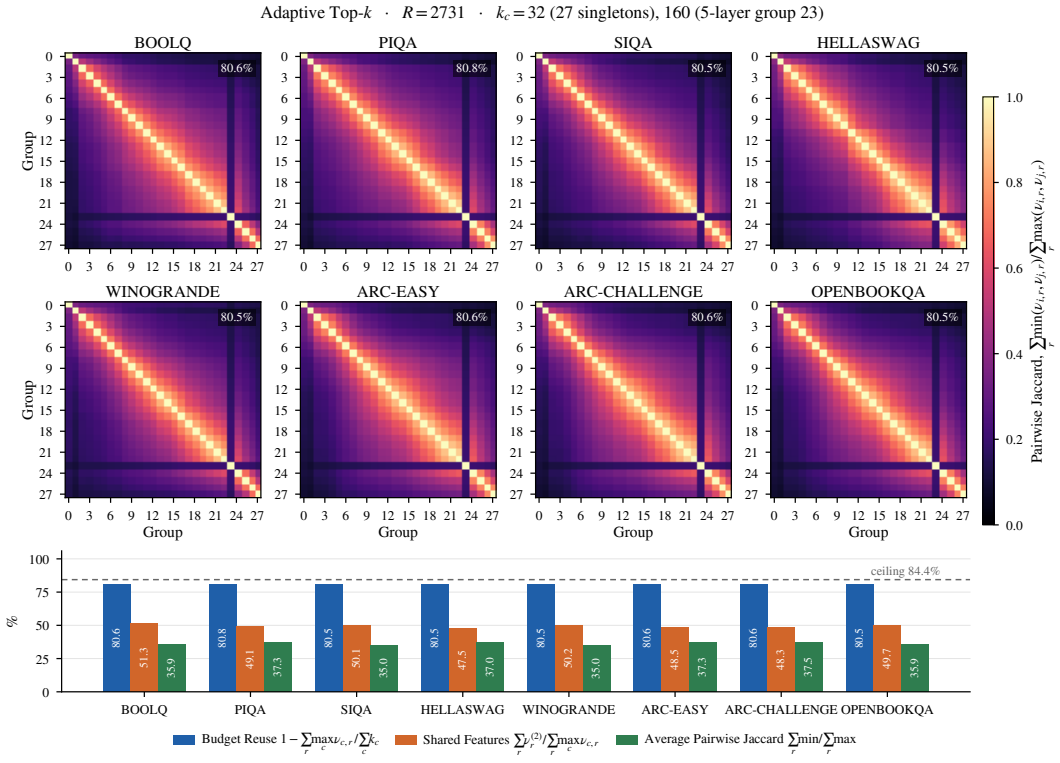

Figure 14: SURF Feature Communication: Adaptive Scheme, $R = 2731$, $\tau = 0.96$, Top-$k = 32$.

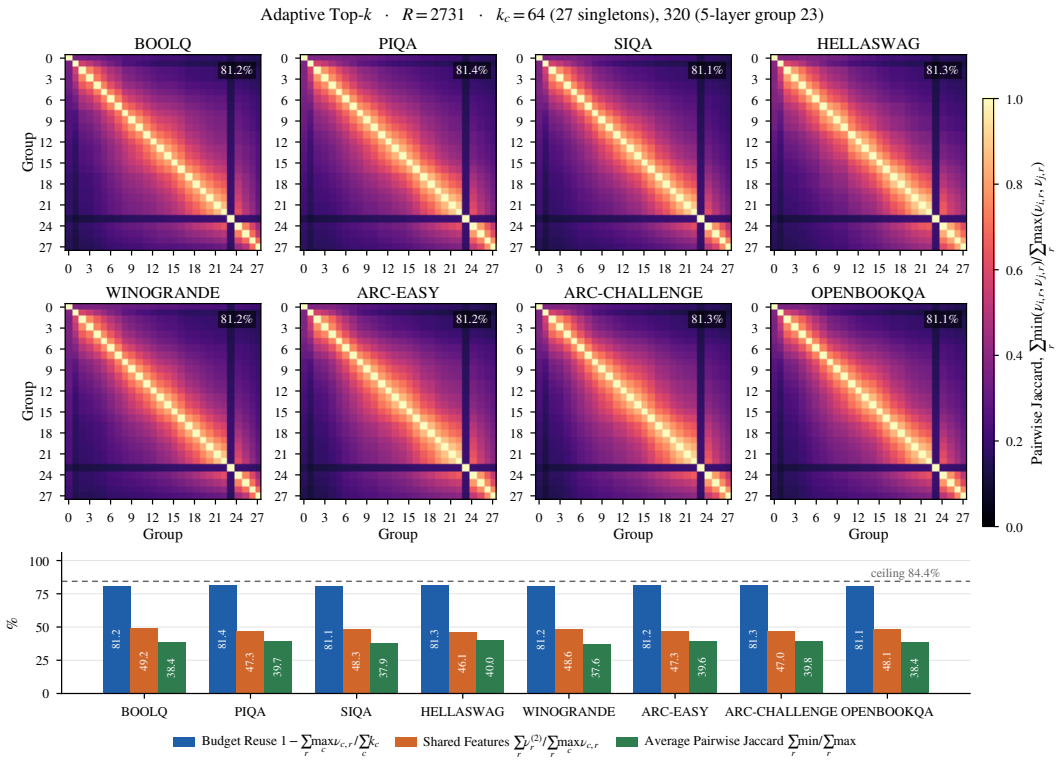

Figure 15: SURF Feature Communication: Adaptive Scheme, $R = 2731$, $\tau = 0.96$, Top-$k = 64$.