Not All Directions Matter: Toward Structured and Task-Aware Low-Rank Adaptation

Anonymous ACL submission

Abstract

Low-Rank Adaptation (LoRA) has become a cornerstone of parameter-efficient fine-tuning (PEFT). Yet, its efficacy is hampered by two fundamental limitations: *semantic drift*, by treating all update directions with equal importance, and *structural incoherence*, from adapting layers independently, resulting in suboptimal, uncoordinated updates. To remedy these, we propose **StructLoRA**, a framework that addresses both limitations through a principled, dual-component design: (1) an Information Bottleneck-guided filter that prunes task-irrelevant directions to mitigate semantic drift, and (2) a lightweight, training-only graph-based coordinator that enforces inter-layer consistency to resolve structural incoherence. Extensive experiments across large language model , vision language model, and vision model (including LLaMA, LLaVA, and ViT) demonstrate that StructLoRA consistently establishes a new state-of-the-art, outperforming not only vanilla LoRA but also advanced dynamic rank allocation and sparsity-based methods. Notably, the benefits are particularly pronounced in challenging low-rank and low-data regimes. Crucially, since our proposed modules operate only during training, StructLoRA enhances performance with **zero additional inference cost**, advancing the focus of PEFT—from mere parameter compression to a more holistic optimization of information quality and structural integrity.

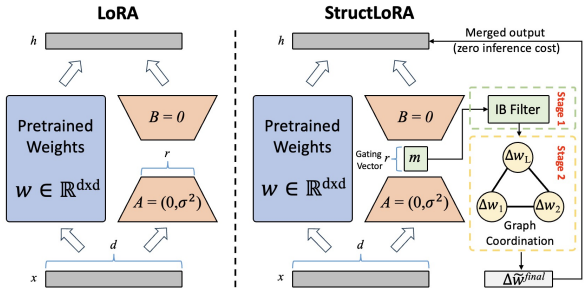

Figure 1: **Architectural comparison between LoRA and StructLoRA.** The left illustrates the standard LoRA architecture with uniform low-rank updates, while the right shows our StructLoRA, which introduces an *Information Bottleneck (IB) filter* and a *Graph-based Coordination mechanism*. These modules selectively retain task-relevant update directions and align layer-wise updates through message passing. Both operate only during training and are removed at inference, preserving LoRA's zero-latency efficiency.

1 Introduction

The advent of large language models (LLMs) has revolutionized natural language processing, yet their immense scale renders full fine-tuning prohibitively expensive for adaptation to downstream tasks. Parameter-Efficient Fine-Tuning (PEFT) has emerged as the *de facto* solution, enabling customization by updating only a small fraction of a model's parameters (Han et al., 2024). Among these methods, Low-Rank Adaptation (LoRA) (Hu et al., 2022a) has become a dominant paradigm due to its elegant simplicity, parameter efficiency, and zero-latency inference (Mao et al., 2025). Its success has spurred a family of variants, such as QLoRA (Dettmers et al., 2023) and AdaLoRA (Zhang et al., 2023b), which further enhance efficiency through quantization and adaptive rank allocation.

However, despite their widespread adoption, we identify two fundamental, unaddressed shortcomings in the LoRA paradigm. The first, *semantic drift*, stems from allocating a limited parameter budget uniformly across all low-rank update directions, implicitly assuming each direction is equally important for the target task . This assumption overlooks the fact that many directions may capture redundant or noisy signals, leading to inefficient use of capacity and suboptimal performance. The second, *structural incoherence*, arises from adapting each layer independently, disregarding the inherent compositional structure of deep models like Transformers (Raghu et al., 2021; Hu et al.,

2022b; Touvron et al., 2022). This layer-wise independence can lead to *semantic drift*, where update directions across layers become misaligned. We empirically observe this phenomenon, finding low cosine similarity between the update gradients of adjacent layers (e.g., 0.27–0.41), which points to a lack of semantic coordination that can hinder generalization. (Sankararaman et al., 2020; Jiang et al., 2024; Carbonnelle and De Vleeschouwer, 2018)

To address these critical gaps, we introduce **StructLoRA**, a novel framework that enhances LoRA with two synergistic components: task-aware filtering and structural coordination. An overview is depicted in Figure 1. To counteract semantic drift, StructLoRA first employs a principled filtering mechanism inspired by the Information Bottleneck principle (Tishby and Zaslavsky, 2015; Alemi et al., 2017). This module learns to preserve only those update directions that are maximally informative for the task objective by optimizing a constrained objective. Subsequently, to mitigate structural incoherence, we explicitly model inter-layer dependencies by representing the network as a computational graph. A lightweight and tailored Graph Neural Network (GNN) then propagates and refines these filtered updates, encouraging smoother and more aligned adaptation trajectories across the model's depth. Crucially, this coordination module operates **only during training** and is discarded at inference, perfectly preserving LoRA's zero-latency advantage.

We empirically validate the effectiveness of StructLoRA across a comprehensive suite of benchmarks, fine-tuning powerful large language model, vision language model, and vision model such as LLaMA, LLaVA, and ViT on challenging tasks in natural language understanding, computer vision, and multimodal reasoning. Our results demonstrate that StructLoRA consistently and significantly outperforms strong PEFT baselines—including LoRA, AdaLoRA, and DoRA—particularly in low-resource and low-rank regimes. Furthermore, StructLoRA substantially closes the performance gap to full fine-tuning while incurring negligible training overhead and zero additional inference latency.

In summary, our contributions are as follows:

- To our knowledge, this work is the first to identify and empirically demonstrate the problem of *semantic drift* and *structural incoherence* in LoRA-based methods, for which we propose a principled solution that combines task-aware filtering and structural coordination.
- We propose a new **StructLoRA**, a lightweight, plug-and-play framework featuring an information bottleneck-guided filter to prune noisy update directions and a graph-based coordinator to enforce inter-layer consistency, which is *detached at inference time*.
- We conduct extensive experiments on large language, multimodal and vision models, showing that StructLoRA sets a new state-of-the-art for parameter-efficient fine-tuning, with particularly strong gains in low-rank and few-shot scenarios.

2 Related Work

2.1 The Landscape of PEFT

Adapting large pre-trained models with limited compute has motivated a rich line of Parameter-Efficient Fine-Tuning (PEFT) methods (Han et al., 2024; Prottasha et al., 2025; Xin et al., 2024; Braz, 2025). Three families are common: additive adapters that insert small trainable blocks (Houlsby et al., 2019; He et al., 2022), selective updates that tune only a subset of native weights such as biases (Ben-Zaken et al., 2022), and reparameterization approaches that learn a low-dimensional proxy for the update. Our work follows the third family and builds on LoRA (Hu et al., 2022a), whose low-rank updates merge into the backbone after training and thus add no inference latency. StructLoRA keeps LoRA's interface but introduces two principles that target overlooked issues in this family: task-aware directional filtering and cross-layer structural coordination.

2.2 The Evolution of LoRA

LoRA (Hu et al., 2022a) has spurred variants along complementary axes. For resource footprint, QLoRA combines LoRA with 4-bit quantization to reduce memory (Dettmers et al., 2023); parameter sharing further cuts trainables (e.g., VeRA and Tied-LoRA) (Kopiczko et al., 2024; Renduchintala et al., 2024). To relax fixed-rank budgets, adaptive schemes allocate capacity by SVD-style importance or learnable rank signals (Zhang et al., 2023b; Shinwari et al., 2025). Sparsity-oriented lines prune modules or preserve sparsity in already sparse models (Drost, 2024; Hu et al., 2025, 2024). Another thread improves update quality: DoRA

disentangles magnitude and direction (Liu et al., 2024b), while orthogonality-based constraints aim to stabilize learning and reduce interference (Qiu et al., 2023; Yuan et al., 2024). These advances decide how much capacity to allocate and how to regularize factors; they rarely ask which directions in the low-rank subspace are semantically useful for the task or how layer-wise updates should be coordinated. StructLoRA addresses both by coupling an Information-Bottleneck filter (Tishby and Zaslavsky, 2015) with a lightweight, training-only coordinator that propagates filtered signals across depth.

3 Methodology

We present **StructLoRA**, a light extension to LoRA that learns *what* directions to keep and *how* layers should move together. The design follows two observations: many low-rank directions carry little signal for the target task, and layer-wise updates often drift when trained in isolation. StructLoRA addresses both with a two-step procedure over the low-rank updates at each layer: an information bottleneck filter that selects task-relevant directions, and a graph-based coordinator that aligns updates across depth.

3.1 Preliminaries: Low-Rank Adaptation

Given a pretrained weight matrix $\mathbf{W}_0 \in \mathbb{R}^{d\times k}$, LoRA learns a rank-$r$ update $\boldsymbol{\Delta W} = \mathbf{AB}$ with $\mathbf{A} \in \mathbb{R}^{d\times r}$ and $\mathbf{B} \in \mathbb{R}^{r\times k}$, while keeping $\mathbf{W}_0$ frozen (Hu et al., 2022a). The forward becomes Eqn (1), and the product $\mathbf{AB}$ can be merged into $\mathbf{W}_0$ after training, so inference cost does not change.

$$\mathbf{y} = \big(\mathbf{W}_0 + \alpha\mathbf{AB}\big)\mathbf{x} \tag{1}$$

StructLoRA keeps this interface and adds training-time selectivity and coordination.

3.2 Stage 1: Information Bottleneck-Guided Directional Filtering

LoRA spreads a small budget over r directions and treats them the same. In practice, only a few directions help predict the label; others carry nuisance variation. We want the update to retain information about $\mathbf{Y}$ but discard information about $\mathbf{X}$ that does not help the task.

Let $\boldsymbol{\Delta W} = \mathbf{AB}$. We gate the r rank-one directions with a learnable mask $\mathbf{m} \in [0,1]^r$ and form the filtered update, as shown in Eqn (2), where $\mathrm{diag}(\mathbf{m})$ denotes a diagonal matrix whose diagonal entries are the elements of $\mathbf{m}$, scaling each rank-one direction by its corresponding gate value.

$$\boldsymbol{\Delta}\tilde{\boldsymbol{W}} = \mathbf{A}\mathrm{diag}(\mathbf{m})\mathbf{B} \tag{2}$$

We learn $\mathbf{m}$ by an information bottleneck (IB) objective (Tishby and Zaslavsky, 2015; Alemi et al., 2017), as described in Eqn (3), where $\mathcal{L}_{\text{task}}$ is the supervised loss, and $I(\cdot;\cdot)$ denotes mutual information.

$$\mathcal{L}_{\text{IB}} = \mathcal{L}_{\text{task}} + \beta\, I\big(\boldsymbol{\Delta}\tilde{\boldsymbol{W}};\, \mathbf{X}\big) - \gamma\, I\big(\boldsymbol{\Delta}\tilde{\boldsymbol{W}};\, \mathbf{Y}\big) \tag{3}$$

Following the variational IB derivation (Alemi et al., 2017), we use a tractable upper bound in which the compression term introduces a KL penalty between a learned posterior over the gate and a simple prior; this acts like a sparsity/weight-decay regularizer on $\mathbf{m}$. When we need hard selection, we use a Gumbel-Softmax relaxation to keep training smooth while pushing $\mathbf{m}$ toward $\{0,1\}$ (Jang et al., 2017). If we estimate MI terms $I(\cdot;\cdot)$ explicitly, we adopt standard variational estimators such as MINE (Belghazi et al., 2018). In short, the filter lets LoRA spend rank on the few directions that move the loss.

Eq. (3) raises the signal-to-noise ratio of the update: it rewards dependence on labels and penalizes spurious dependence on inputs. Under small r or small data, this selectivity reduces variance and makes training steadier. We observe that the filtered updates show higher cross-layer consistency and better task scores than the unfiltered ones (see Section 4).

3.3 Stage 2: Graph-Based Layer Coordination

Backprop couples all parameters through the loss, but it does not impose a structural prior on how *adjacent* layers should update. In Transformers, features change gradually with depth. If layers move in different directions, the trajectory fragments. We add a light coordinator that lets layers exchange their update signals.

Graph construction We view the network as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with one node per layer. The node feature is the flattened filtered update $\mathbf{h}_\ell^{(0)} = \mathrm{vec}(\boldsymbol{\Delta}\tilde{\boldsymbol{W}}_\ell)$. We connect adjacent layers, and we may add semantic edges between layers whose batch-averaged gradients are highly aligned (e.g., by cosine threshold). This captures both depth adjacency and data-driven correlation.

Message passing and reconstruction We run a shallow GNN (GCN or GAT) with a residual path (Kipf and Welling, 2017; Veličković et al., 2018) as described in Eqn (4).

$$\mathbf{h}_\ell^{(t+1)} = \mathbf{h}_\ell^{(t)} + \sigma\left(\sum_{j\in\mathcal{N}(\ell)\cup\{\ell\}} \frac{1}{\sqrt{d_\ell d_j}}\,\mathbf{h}_j^{(t)}\mathbf{\Theta}^{(t)}\right) \tag{4}$$

Here $\mathcal{N}(\ell)$ is the neighbor set, d_ℓ is node degree, $\mathbf{\Theta}^{(t)}$ is a learned matrix, and σ is a nonlinearity. After T steps (we keep T small), we map back to parameter space, as shown in Eqn (5).

$$\mathbf{\Delta}\tilde{\boldsymbol{W}}_\ell^{\text{final}} = \text{reshape}\big(\mathbf{W}_o\mathbf{h}_\ell^{(T)}\big) \tag{5}$$

The coordinator runs only at training time. At inference, we merge $\mathbf{\Delta}\tilde{\boldsymbol{W}}^{\text{final}}$ into $\mathbf{W}_0$, so runtime stays the same as LoRA. For more detailed proof process, please refer to *Appendix* E.

3.4 A Minimal Theoretical View: Coordination as Laplacian Smoothing

We treat the layer-wise update $\mathbf{\Delta}\tilde{\boldsymbol{W}}_\ell$ at depth ℓ as a signal along the network depth and denote its vectorized form as $\mathbf{u}_\ell = \text{vec}(\mathbf{\Delta}\tilde{\boldsymbol{W}}_\ell)$. We then define the inter-layer drift energy as Eqn (6), where $\mathbf{L}$ is the graph Laplacian over depth.

$$\mathcal{E}(\mathbf{U}) = \sum_{\ell=1}^{L-1} \big\|\mathbf{u}_{\ell+1} - \mathbf{u}_\ell\big\|_2^2 = \mathbf{U}^\top(\mathbf{L}\otimes\mathbf{I})\mathbf{U} \tag{6}$$

One residual message-passing step can be written (to first order) as Eqn (7), which equals a gradient step that reduces $\mathcal{E}(\mathbf{U})$.

$$\mathbf{U}^{(t+1)} \approx \mathbf{U}^{(t)} - \eta(\mathbf{L}\otimes\mathbf{I})\mathbf{U}^{(t)} \tag{7}$$

In other words, coordination acts like a Laplacian smoother on update directions. Backprop still optimizes $\mathcal{L}_{\text{task}}$; the coordinator adds an explicit structural prior that lowers drift. These quantities are simple to compute and serve as diagnostics together with adjacent-layer cosine in Section 4. For more detailed proof process, please refer to *Appendix* E.

3.5 Objective, Training, and Inference

We train end-to-end with the loss function described in Eqn (8), where $\mathcal{L}_{\text{IB}}$ is the variational surrogate from Eq. (3).

$$\begin{aligned}\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}}\Big(\mathbf{Y}, f(\mathbf{X};\, \mathbf{W}_0 + \mathbf{\Delta}\tilde{\boldsymbol{W}}^{\text{final}})\Big) \\ + \lambda_{\text{IB}}\,\mathcal{L}_{\text{IB}}(\mathbf{m})\end{aligned} \tag{8}$$

The IB filter and the GNN are *training-only*; both are dropped for inference, so latency stays unchanged. In practice, we use a shallow GNN (1 to 2 layers) and amortize MI estimation across layers to keep overhead modest.

4 Experiments

4.1 Experimental Setup

Models and Architectures Our evaluation employed various powerful and publicly available foundation models, including LLaMA-7B/13B (Touvron et al., 2023), LLaMA3.1-8B (Meta AI, 2024), Qwen2.5-7B (Yang et al., 2024), and Gemma 2 9B (Gemma Team, 2024) for natural language understanding and generation tasks; ViT-B/16 (Dosovitskiy et al., 2021) for image classification; LLaVA-1.5-7B (Liu et al., 2024a) for vision-language instruction following. For all experiments, the base model weights were kept frozen. In Transformer architectures, PEFT methods were applied to the query (W_q) and value (W_v) projection matrices in the self-attention blocks, a standard and effective configuration.

Tasks and Datasets We selected diverse challenging benchmarks to assess performance across different domains. For Natural Language Understanding, we evaluated on the full GLUE benchmark (Wang et al., 2018) and a comprehensive set of eight commonsense reasoning benchmarks: BoolQ, PIQA, HellaSwag, WinoGrande, ARC-e, ARC-c, OBQA, and CSQA. Following standard practice, each task was fine-tuned individually on its respective training set. For Natural Language Generation, we used the Magpie-Pro (Xu et al., 2024) and OpenPlatypus (Lee et al., 2023) datasets to evaluate instruction-following capabilities. For Image Classification, we used ImageNet-1k, CIFAR-100, and the fine-grained Oxford-IIIT Pet dataset. For Multimodal Reasoning, we used established vision-language benchmarks including MS COCO for captioning, VQAv2 and GQA for visual question answering. More details about the baseline, implementation and evaluation can be found in the *Appendix* A.

4.2 Main Results

Our empirical evaluation demonstrates that Struct-LoRA establishes a new state-of-the-art in parameter-efficient fine-tuning, consistently outperforming a wide range of established and recent baselines across diverse modalities. The results

Method	Type	BoolQ	PIQA	CIFAR-100	ImageNet-1k	COCO Caption	VQAv2
Full Fine-tuning	-	82.6	85.3	85.9	78.8	123.5	76.2
Linear Probing	-	71.1	74.8	65.7	68.4	105.3	67.0
Adapter (Houlsby et al., 2019)	Additive	79.2	82.5	81.3	75.7	116.5	73.0
Prefix-Tuning (Li and Liang, 2021)	Additive	79.6	82.8	81.7	76.1	117.0	73.3
LoRA (Hu et al., 2022a)	Reparam.	79.1	82.4	81.5	76.2	116.2	73.5
QLoRA (Dettmers et al., 2023)	Reparam.	80.0	83.1	82.7	76.9	119.1	74.2
DoRA (Liu et al., 2024b)	Reparam.	80.6	83.7	83.2	77.3	120.3	75.0
VeRA (Kopiczko et al., 2024)	Reparam.	79.8	83.0	82.3	76.4	118.4	74.1
DyLoRA (Valipour et al., 2023)	Dynamic Rank	80.1	83.5	82.8	77.0	119.5	74.8
Sensitivity-LoRA (Zhang et al., 2025)	Dynamic Rank	80.9	84.0	83.5	77.5	120.8	75.2
LoRA-Dropout (Lin et al., 2024)	Sparsity	80.2	83.3	82.5	76.8	118.8	74.5
LoRAPrune (Zhang et al., 2023a)	Sparsity	79.8	82.9	82.1	76.6	118.1	74.0
StructLoRA (ours)	**Filtering + Coordination**	**82.1**	**84.9**	**85.1**	**78.6**	**122.9**	**75.9**

Table 1: **Main comparison of StructLoRA with baseline PEFT methods.** We report primary metrics (Accuracy % for classification/reasoning, CIDEr for captioning) under a comparable budget of trainable parameters (∼0.5–1%). See the *Appendix* B for additional results.

Method	MNLI	SST-2	MRPC	CoLA	QNLI	QQP	RTE	STS-B	**Avg.**
LoRA	87.3	93.5	87.1	58.8	93.0	90.5	79.4	91.0	85.1
AdaLoRA	87.3	93.6	87.3	59.0	93.1	90.6	79.6	91.2	85.2
DyLoRA	87.2	93.7	87.3	59.0	93.0	90.6	79.6	91.2	85.2
Sensitivity-LoRA	87.6	94.6	87.7	60.2	93.6	90.7	81.8	91.3	86.0
StructLoRA (ours)	**88.1**	**95.0**	**88.5**	**61.5**	**94.1**	**91.0**	**82.3**	**91.5**	**86.5**

Table 2: **Head-to-head comparison on the GLUE benchmark using RoBERTa-base.** All baseline settings follow Zhang et al. (2025). StructLoRA consistently surpasses all dynamic rank allocation methods, demonstrating superior performance on standard NLU tasks.

validate our core hypotheses: that principled, task-aware directional filtering and structural coordination of updates are critical for unlocking the full potential of low-rank adaptation.

Overall Performance As shown in Table 1, StructLoRA achieves strong performance across all language, vision, and multimodal benchmarks. On the commonsense reasoning tasks BoolQ and PIQA, StructLoRA surpasses the strongest dynamic rank allocation methods, including Sensitivity-LoRA and DyLoRA, by a significant margin. The performance gains are even more pronounced on vision tasks, StructLoRA outperforms all baselines on CIFAR-100 and ImageNet-1k, achieving results that are nearly on par with full fine-tuning while using less than 1% of the trainable parameters.

StructLoRA also demonstrates superiority over methods that employ heuristic sparsity. While LoRA-Dropout (Lin et al., 2024), a strong regularization baseline, improves upon vanilla LoRA, it still falls short of StructLoRA's performance. This suggests that the principled, task-aware filtering of our Information Bottleneck module is more effective at identifying and pruning irrelevant update directions than untargeted, stochastic dropout. Similarly, StructLoRA outperforms structured pruning methods like LoRAPrune (Zhang et al., 2023a), indicating that dynamic, in-training coordination and filtering is more effective than iterative, gradient-based pruning schemes.

Head-to-Head on the GLUE Benchmark To provide a direct, controlled comparison with recent dynamic rank allocation methods, we evaluate StructLoRA on the GLUE benchmark using the same RoBERTa-base setup as Zhang et al. (2025). As shown in Table 2, StructLoRA achieves a new state-of-the-art average score of **86.5**, outperforming Sensitivity-LoRA by a margin of 0.5 points. This is particularly noteworthy as Sensitivity-LoRA is highly optimized for this benchmark. The superior performance of StructLoRA suggests that its dual mechanism of filtering irrelevant information and enforcing structural coherence provides a more robust and effective adaptation strategy than relying solely on parameter sensitivity heuristics.

Superior Efficiency in Low-Rank Regimes A key test for any PEFT method is its ability to perform under highly constrained parameter budgets. Table 3 evaluates StructLoRA's performance as a function of the LoRA rank r. The results clearly

Rank (r)	Trainable Params (%)	BoolQ (LLaMA-7B)		CIFAR-100 (ViT-B/16)		COCO Caption (LLaVA-1.5-7B)	
		LoRA	**StructLoRA**	LoRA	**StructLoRA**	LoRA	**StructLoRA**
2	0.12	75.1	**77.4** (+2.3)	78.3	**80.1** (+1.8)	111.2	**114.3** (+3.1)
4	0.24	77.6	**79.9** (+2.3)	79.7	**82.2** (+2.5)	113.8	**117.0** (+3.2)
8	0.48	79.1	**81.3** (+2.2)	81.5	**84.1** (+2.6)	116.2	**122.4** (+6.2)
16	0.95	80.3	**81.7** (+1.4)	82.8	**84.3** (+1.5)	118.1	**123.6** (+5.5)
32	1.90	81.0	**81.9** (+0.9)	83.4	**84.5** (+1.1)	119.0	**123.9** (+4.9)

Table 3: **Performance under varying rank budgets.** Accuracy (for BoolQ, CIFAR-100) and CIDEr (for COCO Caption) are reported. StructLoRA consistently outperforms LoRA, with the largest gains observed in low-rank ($r \leq 8$) settings, demonstrating superior parameter efficiency.

Dataset	Metric	Method	10% Data	25% Data	50% Data	100% Data
BoolQ (LLaMA-7B)	Accuracy	LoRA	68.5	73.2	76.4	79.1
		StructLoRA	**71.2** (+2.7)	**76.3** (+3.1)	**78.9** (+2.5)	**81.3** (+2.2)
CIFAR-100 (ViT-B/16)	Accuracy	LoRA	73.6	78.0	80.5	81.5
		StructLoRA	**76.3** (+2.7)	**80.5** (+2.5)	**82.4** (+1.9)	**84.1** (+2.6)
COCO Caption (LLaVA-1.5-7B)	CIDEr	LoRA	100.2	108.3	114.0	116.2
		StructLoRA	**103.7** (+3.5)	**112.4** (+4.1)	**117.9** (+3.9)	**122.4** (+6.2)

Table 4: **Performance under limited supervision (few-shot learning).** We evaluate with a fixed rank $r = 8$ while varying data availability. StructLoRA's advantage over LoRA grows as the amount of training data decreases, highlighting its superior data efficiency and robustness to overfitting.

show that StructLoRA's advantage is most pronounced in low-rank settings. At an aggressive rank of $r = 2$ (representing just 0.12% of the model's parameters), StructLoRA outperforms LoRA by a significant margin of **+2.3%** on BoolQ and **+3.1** CIDEr on COCO Captioning. This demonstrates the critical importance of our IB-guided filter, which acts as an intelligent resource manager. When the parameter budget is scarce, the filter ensures that capacity is allocated only to the most task-relevant update directions, preventing the model from wasting parameters on noisy or redundant signals. As the rank increases, the performance gap narrows but remains consistently in favor of StructLoRA, indicating that even with a larger budget, the combination of filtering and structural coordination provides a superior inductive bias for adaptation.

Robustness in Low-Data Regimes In many real-world applications, labeled data is scarce. We evaluate StructLoRA's performance when fine-tuned on fractions of the full training data. Table 4 shows that StructLoRA is significantly more robust than LoRA in low-data settings. With only 10% of the training data, StructLoRA's performance advantage widens to **+2.7%** on BoolQ and a remarkable **+3.5** CIDEr on COCO Captioning. This heightened robustness stems from the dual regularizing effects of our framework. The IB-filter prevents the model from overfitting to spurious correlations in the small dataset by pruning noisy directions, while the GNN coordinator enforces a structural prior that encourages smoother, more generalizable updates across layers. This demonstrates that StructLoRA is not only more parameter-efficient but also more data-efficient.

4.3 Ablation Studies

To validate the design of StructLoRA and disentangle the contributions of its core components, we conduct a series of rigorous ablation studies. Our analysis is structured to answer three key questions: (1) *Are both the IB filter and the GNN coordinator necessary for the observed performance gains?* (2) *Is our specific design of the GNN coordinator optimal?* (3) *Is the information-theoretic filtering strategy superior to simpler heuristics?*

Contribution of Core Components As shown in Table 5, removing either the Information Bottleneck (IB) filter or the Graph Neural Network (GNN) coordinator leads to notable performance degradation across all tasks, while removing both reduces StructLoRA to standard LoRA. In particular, removing the IB filter causes the largest drop (e.g., **-1.9%** on BoolQ and **-4.6** CIDEr on COCO Captioning), highlighting the importance of filtering out task-irrelevant update directions. This supports our central hypothesis that much of the

Ablation Setting	BoolQ (LLaMA-7B)		CIFAR-100 (ViT-B/16)		COCO Caption (LLaVA-1.5-7B)	
	Accuracy (%)	Δ	Accuracy (%)	Δ	CIDEr	Δ
StructLoRA (Full)	**81.3**	0.0	**84.1**	0.0	**122.4**	0.0
w/o IB Filter	79.4	-1.9	81.9	-2.2	117.8	-4.6
w/o GNN Coordination	80.1	-1.2	82.6	-1.5	119.4	-3.0
w/o Both (i.e., Standard LoRA)	79.1	-2.2	81.5	-2.6	116.2	-6.2

Table 5: **Module-wise ablation of StructLoRA.** Removing either *information bottleneck (IB)* filter or the *graph-based coordination network* degrades performance, while removing both collapses StructLoRA to LoRA, demonstrating their synergistic contribution. See further analysis in *Appendix* C.

low-rank subspace in vanilla LoRA is occupied by noisy or redundant information. The GNN coordinator also yields consistent gains (e.g., **-1.2%** on BoolQ and **-1.5%** on CIFAR-100), confirming that enforcing structural coherence across layers is crucial for generalization. Together, the two modules act synergistically: the IB filter produces cleaner, task-relevant signals, which the GNN coordinator effectively propagates across layers.

Effect of the GNN Coordinator Design A key criticism of prior work is that coordination mechanisms can be arbitrary. We therefore conduct targeted ablations to justify our specific design choices for the GNN module. As shown in Table 6, we find that a shallow, 1-layer GNN provides the optimal balance. Increasing the depth to 2 or 3 layers leads to a consistent decline in performance, likely due to oversmoothing, a known issue in deep GNNs where node representations become indistinguishable (Kipf and Welling, 2017). We also find that our hybrid graph construction, which uses both structural adjacency and semantic similarity for edges, outperforms simpler graphs that use only one of these criteria. This confirms that our GNN design is not arbitrary but is a carefully considered choice that provides robust, lightweight coordination without falling into common GNN pitfalls.

GNN Configuration	Accuracy (%)
StructLoRA	**81.3**
Effect of Depth:	
2-Layer GNN	80.4
3-Layer GNN	79.7
Effect of Graph Construction:	
Adjacency Only	80.5
Similarity Only	80.2

Table 6: **Ablation on GNN design choices on the BoolQ dataset.** The default StructLoRA employs a 1-layer GNN with a hybrid graph combining adjacency and similarity edges, achieving the highest accuracy.

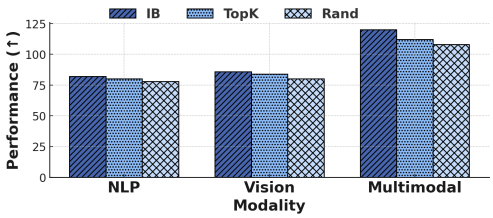

Figure 2: **Analysis of filtering strategies.** We compare our IB-guided filter with two heuristics under the same keep ratio: *Random Masking* and *Top-k Norm* (scored by $\|a_j\|_2\|b_j\|_2$ for each rank-one direction). Bars show mean performance across three random seeds; error bars indicate 95% confidence intervals.

Effect of Filtering Strategies To validate our use of a principled, information-theoretic filter, we compare it against several simpler heuristic-based filtering strategies in Figure 2. We compare our IB-guided filtering with (1) Random Masking, where a random subset of directions are kept; and (2) Top-k Norm, where the directions corresponding to the largest L2-norms in the update matrix are retained. Our IB-guided approach consistently and significantly outperforms all alternatives across all modalities. Notably, while norm-based filtering provides a slight improvement over random selection, it is substantially worse than our method. This result provides strong evidence that the magnitude of an update direction is a poor proxy for its semantic relevance to the task, a key insight that motivates our work.

Analysis of Training Overhead We analyze the computational overhead introduced by StructLoRA during training. As shown in Table 7, our framework introduces a negligible increase in training time and peak memory usage compared to standard LoRA. On LLaMA-7B, StructLoRA is only **4-6%** slower per epoch and requires less than **0.8 GB** of additional peak GPU memory. This minimal overhead is a small price for the significant performance gains, and importantly, this cost is in-

curred **only during training**, as both the IB filter and GNN coordinator are discarded at inference time, preserving LoRA's zero-latency advantage.

Method	Training Time / Epoch	Peak Memory (GB)
LoRA	1.00×	16.8
StructLoRA	1.06×	17.5

Table 7: **Training overhead analysis on LLaMA-7B with rank $r = 8$.** StructLoRA introduces minimal overhead compared to standard LoRA.

4.4 Visualizing Structural Coherence

Beyond quantitative metrics, we conduct a series of qualitative analyses to visually inspect *how* StructLoRA alters the model's adaptation dynamics. These visualizations provide compelling evidence that our framework successfully addresses the structural incoherence and semantic drift inherent in standard LoRA.

Attention Alignment Analysis Figure 3 shows Grad-CAM (Selvaraju et al., 2017) visualizations on ViT-B/16 for two ImageNet samples. The upper row corresponds to LoRA, while the lower row shows StructLoRA. LoRA produces diffuse attention that spreads across background regions, whereas StructLoRA focuses on task-relevant areas such as the deer's head or the boat's body. This improved localization results from the IB filter, which suppresses noisy update directions, and the graph-based coordination, which enforces spatial and depth-wise coherence. Together, these mechanisms yield sharper, more discriminative feature maps and stronger alignment between attention and semantic content, highlighting StructLoRA's *semantic consistency* and *structural awareness*.

Mitigating Semantic Drift with Inter-Layer Coordination We analyze the effect of the GNN coordinator by measuring cosine similarity between the update directions of adjacent layers. Figure 4 shows that LoRA's updates are noisy and inconsistent, with low similarity across layers, confirming the presence of *semantic drift*. In contrast, StructLoRA produces a clear block-diagonal pattern in the similarity heatmap. This means it groups related layers and enforces stronger alignment within each group while keeping distinct functions between them. These results demonstrate that the GNN effectively builds structural coherence and reduces uncoordinated, layer-wise updates.

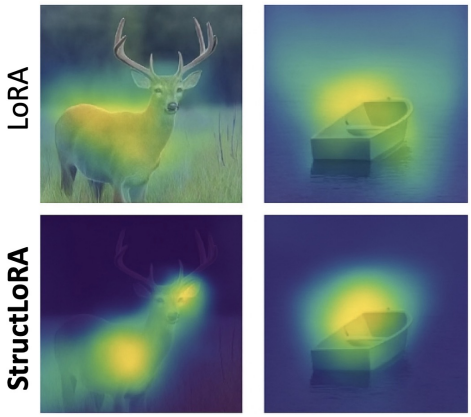

Figure 3: **Visual attention comparison between LoRA and StructLoRA.** The top row shows Grad-CAM (Selvaraju et al., 2017) heatmaps from the baseline LoRA model, while the bottom row corresponds to StructLoRA. StructLoRA produces more concentrated and semantically aligned activation regions.

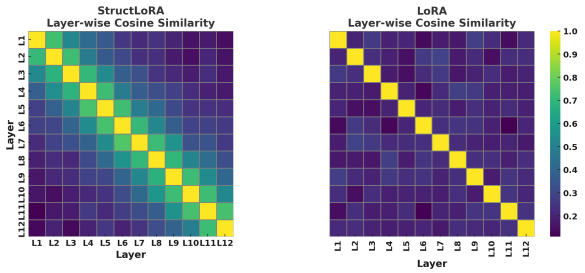

Figure 4: **Layer-wise cosine similarity of updates.** StructLoRA induces a coherent block-diagonal structure, while LoRA exhibits noisy and fragmented activation patterns. See Table 13 for more results.

5 Conclusion

In this work, we introduced StructLoRA, a principled framework tackles two fundamental limitations of low-rank adaptation: semantic drift and structural incoherence, by combining an Information Bottleneck-based filter for task-relevant updates with a graph-based coordinator for cross-layer consistency. Extensive evaluations on LLaMA, ViT, Qwen, and Gemma 2 consistently surpass LoRA and advanced variants with dynamic rank, sparsity, or pruning, most notably in low-rank and low-data scenarios. Ablation studies and visualizing structural coherence analysis show that StructLoRA mitigates semantic drift. Beyond efficiency, StructLoRA also redefines parameter tuning as a joint optimization of information and structure. Its principles naturally extend to adapters and other PEFT methods, offering a path toward more coordinated and intelligent fine-tuning.

Limitations

While StructLoRA demonstrates strong empirical performance and introduces a novel, principled approach to PEFT, we acknowledge several limitations that present avenues for future work. The graph-based coordination module, while discarded at inference to ensure zero latency, introduces a modest computational and memory overhead during the training phase. Although we find this overhead to be minimal in our experiments (see Section 4.3), it may become a more significant consideration when scaling to models with hundreds of layers or in highly resource-constrained training environments.

References

Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. 2017. Deep variational information bottleneck. In *International Conference on Learning Representations (ICLR)*.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. 2018. Mutual information neural estimation. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pages 531–540. PMLR.

Elad Ben-Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1–9.

Samuel Lima Braz. 2025. Peft: Parameter-efficient fine-tuning methods for llms. Hugging Face Blog.

Simon Carbonnelle and Christophe De Vleeschouwer. 2018. Layer rotation: a surprisingly powerful indicator of generalization in deep networks? *arXiv preprint arXiv:1806.01603*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. In *Advances in Neural Information Processing Systems (NeurIPS)*.

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. 2021. An image is worth 16×16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*.

Dorian Drost. 2024. An overview of the lora family. Medium.

Gemma Team. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Zeyu Han, Chao Gao, Jinyang Liu, Jeff Zhang, and Sai Qian Zhang. 2024. Parameter-efficient fine-tuning for large models: A comprehensive survey. *arXiv preprint arXiv:2403.14608*.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations (ICLR)*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. *arXiv preprint arXiv:1902.00751*.

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

Jinyi Hu, Xiaoyuan Yi, Wenhao Li, Maosong Sun, and Xing Xie. 2022b. Fuse it more deeply! a variational transformer with layer-wise latent variable inference for text generation. In *Proceedings of NAACL-HLT 2022*, pages 697–716.

Yuxuan Hu, Jing Zhang, Zhe Zhao, Cuiping Li, and Hong Chen. 2024. Sp-lora: Sparsity-preserved low-rank adaptation for sparse large language models. *OpenReview (ICLR 2025; withdrawn submission)*.

Yuxuan Hu, Jing Zhang, Zhe Zhao, Cuiping Li, and Hong Chen. 2025. Lors: Efficient low-rank adaptation for sparse large language model. *arXiv preprint arXiv:2501.08582*.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations (ICLR)*.

Jiachen Jiang, Jinxin Zhou, and Zhihui Zhu. 2024. Tracing representation progression: Analyzing and enhancing layer-wise similarity. *arXiv preprint arXiv:2406.14479*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Dawid J. Kopiczko, Tijmen Blankevoort, and Yuki M. Asano. 2024. Vera: Vector-based random matrix adaptation. In *International Conference on Learning Representations (ICLR)*.

Ariel N. Lee, Cole J. Hunter, and Nataniel Ruiz. 2023. Platypus: Quick, cheap, and powerful refinement of llms. *arXiv preprint arXiv:2308.07317*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 4582–4597.

Yang Lin, Xinyu Ma, Xu Chu, Yujie Jin, Zhibang Yang, Yasha Wang, and Hong Mei. 2024. Lora dropout as a sparsity regularizer for overfitting control. *Preprint*, arXiv:2404.09610.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024a. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. 2024b. DoRA: Weight-decomposed low-rank adaptation. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pages 32100–32121. PMLR.

Yuren Mao, Yuhang Ge, Yijiang Fan, Wenyi Xu, Yu Mi, Zhonghao Hu, and Yunjun Gao. 2025. A survey on lora of large language models. *Frontiers of Computer Science*, 19(7):197605.

Meta AI. 2024. Introducing llama 3.1: Our most capable models to date. Blog post.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. Adapterfusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 487–503.

Nusrat Jahan Prottasha, Upama Roy Chowdhury, Shetu Mohanto, Tasfia Nuzhat, Abdullah As Sami, Md Shamol Ali, Md Shohanur Islam Sobuj, Hafijur Raman, Md Kowsher, and Ozlem Ozmen Garibay. 2025. Peft a2z: Parameter-efficient fine-tuning survey for large language and vision models. *arXiv preprint arXiv:2504.14117*.

Zeju Qiu, Yutong Liu, Weijia Wen, Li Xu, Yixiao Zhang, Jian Yang, Tong Lu, and Ying Shan. 2023. Controlling text-to-image diffusion by orthogonal finetuning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. 2021. Do vision transformers see like convolutional neural networks? In *Advances in Neural Information Processing Systems (NeurIPS)*.

Adithya Renduchintala, Tugrul Konuk, and Oleksii Kuchaiev. 2024. Tied-lora: Enhancing parameter efficiency of lora with weight tying. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 8694–8705, Mexico City, Mexico. Association for Computational Linguistics.

Karthik Abinav Sankararaman, Soham De, Zheng Xu, W Ronny Huang, and Tom Goldstein. 2020. The impact of neural network overparameterization on gradient confusion and stochastic gradient descent. In *International conference on machine learning*, pages 8469–8479. PMLR.

Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 618–626.

H. U. K. Shinwari and 1 others. 2025. Ard-lora: Dynamic rank allocation for parameter-efficient fine-tuning of large models. *arXiv preprint arXiv:2506.18267*.

Naftali Tishby and Noga Zaslavsky. 2015. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pages 1–5. IEEE.

Hugo Touvron, Matthieu Cord, Alaaeldin El-Nouby, Jakob Verbeek, and Hervé Jégou. 2022. Three things everyone should know about vision transformers. In *Computer Vision – ECCV 2022*, volume 13684 of *Lecture Notes in Computer Science*, pages 497–515. Springer.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Gabriel Rémi, Naman Goyal, Eric Hambro, Faisal Azhar, and 1 others. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. 2023. Dylora: Parameter-efficient tuning of pretrained models using dynamic search-free low-rank adaptation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 3294–3310.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations (ICLR)*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP*, pages 353–355.

Yi Xin, Siqi Luo, Haodi Zhou, Junlong Du, Xiaohong Liu, Yue Fan, Qing Li, and Yuntao Du. 2024. Parameter-efficient fine-tuning for pre-trained vision models: A survey. *arXiv preprint arXiv:2402.02242*.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. 2024. Magpie: Alignment data synthesis from scratch by prompting aligned llms with nothing. *Preprint*, arXiv:2406.08464.

An Yang, Beichen Zhang, Binyuan Hui, Yang Fan, Xingzhang Ren, Jingren Zhou, Junyang Lin, and 1 others. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

Shuo Yuan, Yifan Sun, Ming Jiang, Xinyu Zhang, Zhipeng Liu, Zhiqiang Li, and Ming-Hsuan Xu. 2024. Householder reflection adaptation: Bridging the gap between low-rank and orthogonal adaptation. *arXiv preprint arXiv:2405.17484*.

Hao Zhang, Bo Huang, Zhenjia Li, Xi Xiao, Hui Yi Leong, Zumeng Zhang, Xinwei Long, Tianyang Wang, and Hao Xu. 2025. Sensitivity-lora: Low-load sensitivity-based fine-tuning for large language models. *arXiv preprint arXiv:2509.09119*.

Mingyang Zhang, Hao Chen, Chunhua Shen, Zhen Yang, Linlin Ou, Xinyi Yu, and Bohan Zhuang. 2023a. Loraprune: Structured pruning meets low-rank parameter-efficient fine-tuning. *arXiv preprint arXiv:2305.18403*.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2023b. Adaptive budget allocation for parameter-efficient fine-tuning. In *International Conference on Learning Representations (ICLR)*.

Appendix

This appendix contains additional theoretical proofs, extended experiments, and qualitative analyses supporting the results in our main paper.

A Experimental Setup

Models and Architectures. To verify that StructLoRA generalizes across model scales and modalities, we evaluate on a broad set of foundation models. For natural language understanding and generation, we use LLaMA-7B/13B (Touvron et al., 2023), LLaMA3.1-8B (Meta AI, 2024), Qwen2.5-7B (Yang et al., 2024), and Gemma 2 9B (Gemma Team, 2024). For vision, we employ ViT-B/16 (Dosovitskiy et al., 2021), and for multimodal reasoning, LLaVA-1.5-7B (Liu et al., 2024a). All base weights remain frozen during fine-tuning. Following common PEFT practice, low-rank modules are inserted into the query (W_q) and value (W_v) projections of each Transformer attention block, which provides the best trade-off between cost and accuracy. Unless otherwise noted, rank r is fixed at 8 and the scaling factor α is set to 16.

Tasks and Datasets. We select tasks to cover three complementary domains:

- **Natural Language Understanding.** We fine-tune each model on all GLUE tasks as well as eight commonsense reasoning benchmarks-BoolQ, PIQA, HellaSwag, WinoGrande, ARC-e, ARC-c, OBQA, and CSQA-following the standard single-task protocol. Accuracy is reported on each dataset and the average across tasks.

- **Natural Language Generation.** We use Magpie-Pro (Xu et al., 2024) and OpenPlatypus (Lee et al., 2023) to evaluate instruction-following and general text generation. Evaluation metrics include BLEU-4, ROUGE-L, and perplexity.

- **Vision and Multimodal Tasks.** For vision, we evaluate on ImageNet-1K, CIFAR-100, and Oxford-IIIT Pet. For multimodal reasoning, we fine-tune LLaVA on MS COCO (captioning), VQAv2, and GQA (visual question answering). Metrics include classification accuracy, CIDEr, and VQA accuracy.

All text inputs are tokenized using each model's native tokenizer, and visual inputs are resized to 224×224 with standard augmentations. Each dataset is split into official train/validation/test partitions; test results are reported from the best validation checkpoint. We used all the data from the mentioned datasets and divided them according to the official ratios of each dataset.

Baselines. We compare StructLoRA against a carefully selected set of strong and representative PEFT methods, grouped by their core mechanism, to ensure a thorough and fair evaluation. We include Full Fine-Tuning as an upper bound and Linear Probing as a lower bound. We also compare against classic PEFT methods like Adapters (Houlsby et al., 2019) and Prefix-Tuning (Li and Liang, 2021). Our primary comparison set includes state-of-the-art LoRA variants: vanilla LoRA (Hu et al., 2022a), QLoRA (Dettmers et al., 2023) for memory efficiency, and DoRA (Liu et al., 2024b) for disentangled magnitude and direction updates. Dynamic Rank Allocation Methods: To compare against methods that also address LoRA's uniform budget allocation, we include: AdaLoRA (Zhang et al., 2023b), which uses SVD-based importance; DyLoRA (Valipour et al., 2023), which trains across a range of ranks; and the recent Sensitivity-LoRA (Zhang et al., 2025), which uses Hessian-based metrics. Sparsity and Pruning Baselines: To test our hypothesis against alternative methods for eliminating redundant parameters, we include two additional strong baselines: LoRA-Dropout (Lin et al., 2024), which applies dropout as a heuristic for sparsity regularization, and LoRAPrune (Zhang et al., 2023a), a representative structured pruning method. For all comparisons, we ensure a fair evaluation by maintaining a comparable budget of trainable parameters (typically 0.5–1% of the full model size) across all PEFT methods.

Implementation and Hyperparameters. All experiments are implemented in PyTorch 2.2 and trained on NVIDIA A100 80GB GPUs. Optimization uses AdamW with β_1=0.9, β_2=0.999, weight decay 0.01, and cosine learning-rate decay. Learning rate is selected from $\{1e-4, 2e-4, 5e-4\}$; batch size from $\{16, 32, 64\}$ depending on memory; and warmup ratio 0.06. Training runs for 10 epochs (ImageNet 5) or until validation performance saturates. All random seeds are fixed for reproducibility.

Evaluation and Metrics. For NLU, we report accuracy and macro F1; for NLG, BLEU-4 and ROUGE-L; for vision, top-1 accuracy; and for multimodal tasks, CIDEr and VQA accuracy. All results are averaged over three seeds. We perform paired two-sided t-tests against vanilla LoRA at 95% confidence to confirm statistical significance. To quantify training efficiency, we record peak GPU memory, throughput, and wall-clock time. StructLoRA adds $\approx 6\%$ training overhead but preserves LoRA's zero inference cost.

B Generalization Across Diverse Architectures

A crucial test for any foundational PEFT method is its ability to generalize beyond a single model family. To demonstrate that StructLoRA's principles are model-agnostic and universally applicable, we conduct a rigorous head-to-head comparison on complex Natural Language Generation (NLG) tasks. We evaluate performance on the Magpie-Pro and OpenPlatypus datasets, aligning our setup with recent work (Zhang et al., 2025) for a direct comparison against strong baselines. Our evaluation spans three distinct, state-of-the-art open-source models: **Qwen2.5-7B**, **LLaMA3.1-8B**, and Google's **Gemma 2 9B**.

The results, presented in Table 8, unequivocally establish StructLoRA as the new state-of-the-art for parameter-efficient fine-tuning in the generative domain. Across all three modern architectures and on both datasets, StructLoRA consistently and significantly outperforms every other method, including the highly optimized **Sensitivity-LoRA**.

On LLaMA3.1-8B, StructLoRA surpasses Sensitivity-LoRA by an average of **0.91 points**. This performance gap is maintained across other architectures, with StructLoRA leading by **0.83 points** on Qwen2.5-7B and **0.77 points** on Gemma 2 9B. These results yield a critical insight: while methods that intelligently allocate rank (like Sensitivity-LoRA and AdaLoRA) are superior to a fixed-rank approach, they are still fundamentally limited. Their focus remains on *how much* capacity to assign to each layer. StructLoRA's superior performance stems from its more holistic approach, which addresses two orthogonal problems: it not only prunes noisy and irrelevant information at the sub-layer, directional level via its IB-filter but also ensures that the resulting high-quality update signals are propagated coherently across the entire model via its GNN coordinator.

This dual mechanism of **semantic filtering** and **structural coordination** provides a more robust and effective inductive bias for adaptation than methods that rely solely on parameter sensitivity or rank allocation heuristics. The consistent success of StructLoRA across the LLaMA, Qwen, and Gemma architectures-each with its own unique design nuances-provides powerful evidence of our framework's model-agnostic nature and the universal applicability of its underlying principles.

C Coordination Alternatives: Simple Regularizers vs. Graph Coordination

Set-up. To verify that StructLoRA's coordination gain does not merely come from adding any depth-wise regularization, we compare it against two simpler alternatives under the same LoRA rank and training protocol: **LoRA+Cosine Regularization** (LoRA+Cos): penalizes dissimilar directions between adjacent layers; **LoRA+Laplacian Tikhonov** (LoRA+Lap): minimizes depth-wise drift via a fixed Laplacian quadratic penalty; **StructLoRA (GNN)** (Ours): performs learned, data-aware message passing during training.

Regularizers. Let $\mathbf{u}_\ell = \mathrm{vec}(\boldsymbol{\Delta}\tilde{\boldsymbol{W}}_\ell)$ be the flattened update of layer ℓ. Cosine and Laplacian penalties are defined as:

$$\mathcal{L}_{\text{cos}} = \sum_{\ell=1}^{L-1} \big(1 - \cos(\mathbf{u}_\ell, \mathbf{u}_{\ell+1})\big), \tag{9}$$

$$\mathcal{L}_{\text{lap}} = \sum_{\ell=1}^{L-1} \|\mathbf{u}_{\ell+1} - \mathbf{u}_\ell\|_2^2. \tag{10}$$

Both are added to the task loss with tuned coefficients λ_{cos} or λ_{lap}. StructLoRA instead learns adaptive, data-dependent weights through the GNN.

Evaluation metrics. We report three quantities: (1) **Task Score** (accuracy or CIDEr); (2) **Inter-layer Drift Energy** $\mathcal{E} = \sum_\ell \|\mathbf{u}_{\ell+1} - \mathbf{u}_\ell\|^2$ (↓ better); (3) **Adjacent-layer Cosine** $\mathrm{CosAdj} = \frac{1}{L-1}\sum_\ell \cos(\mathbf{u}_\ell, \mathbf{u}_{\ell+1})$ (↑ better). All variants share identical seeds, optimizer, and steps.

Findings. Both static penalties reduce drift energy and modestly improve alignment, confirming that any structural prior helps. However, StructLoRA achieves the largest gains on all metrics with

Model	Method	Magpie-Pro			OpenPlatypus			Avg.
		BLEU-4	ROUGE-1	ROUGE-L	BLEU-4	ROUGE-1	ROUGE-L	
Qwen2.5-7B	HAdapter	54.71	49.11	32.42	19.39	43.95	22.51	37.01
	PAdapter	54.83	49.15	32.24	19.42	44.03	22.54	37.04
	LoRA	55.03	48.82	32.42	19.72	43.83	22.53	37.06
	AdaLoRA	55.66	49.13	32.75	19.87	44.24	22.67	37.39
	DyLoRA	55.59	49.21	32.82	19.86	44.18	22.59	37.37
	Sensitivity-LoRA	56.31	50.04	33.57	20.13	44.77	23.07	37.98
	StructLoRA (ours)	**56.95**	**50.82**	**34.15**	**20.41**	**45.33**	**23.55**	**38.54**
LLaMA3.1-8B	HAdapter	69.28	56.23	41.08	34.73	52.31	35.61	48.20
	PAdapter	69.30	55.05	41.97	34.66	51.47	36.01	48.07
	LoRA	69.67	55.89	41.78	34.64	52.35	35.92	48.37
	AdaLoRA	70.40	56.31	42.17	34.86	52.90	36.15	48.80
	DyLoRA	70.36	56.26	42.16	34.89	52.80	36.20	48.78
	Sensitivity-LoRA	71.25	57.35	43.02	35.30	53.79	36.69	49.57
	StructLoRA (ours)	**71.98**	**58.02**	**43.96**	**35.85**	**54.51**	**37.41**	**50.29**
Gemma 2 9B	HAdapter	68.85	55.82	40.55	34.11	51.88	35.02	47.71
	PAdapter	68.91	54.77	41.03	34.02	51.05	35.25	47.51
	LoRA	69.15	55.31	40.99	34.20	51.95	35.30	47.82
	AdaLoRA	69.88	55.90	41.45	34.51	52.60	35.77	48.35
	DyLoRA	69.81	55.82	41.40	34.49	52.45	35.70	48.28
	Sensitivity-LoRA	70.65	56.88	42.44	35.01	53.21	36.22	49.07
	StructLoRA (ours)	**71.43**	**57.60**	**43.35**	**35.58**	**54.05**	**37.01**	**49.84**

Table 8: **Evaluation results on Natural Language Generation (NLG) tasks across diverse, state-of-the-art model architectures.** We compare StructLoRA with other PEFT baselines on two representative datasets, Magpie-Pro and OpenPlatypus. StructLoRA consistently establishes a new state-of-the-art, outperforming all other methods-including the strong Sensitivity-LoRA baseline-across all tested models. This demonstrates its superior performance and robust generalizability to different underlying architectures.

Method	Task Score (↑)	$\mathcal{E} \times 10^{-2}$ (↓)	CosAdj (↑)	Extra Cost (train, %)
LoRA+Cos	82.3 / 79.4	3.98 / 4.22	0.46 / 0.42	+1.2
LoRA+Lap	83.0 / 80.1	3.72 / 3.95	0.49 / 0.45	+2.4
StructLoRA (GNN)	**84.1 / 81.3**	**3.05 / 3.41**	**0.56 / 0.52**	+5.3

Table 9: **Coordination alternatives under equal LoRA budget.** Averaged over three runs on ViT-B/16 (CIFAR-100) and LLaMA-7B (BoolQ). $\mathcal{E}$ and CosAdj follow the definitions above.

only a small additional cost. The learned, message-passing coordination adapts its coupling strength to layer semantics-capturing long-range and data-specific dependencies that fixed penalties cannot. This supports our claim that *explicit, learnable coordination* is superior to static regularization and that the observed improvements arise from structured, task-aware smoothing rather than incidental regularization.

D Scalability and Performance in Challenging Scenarios

A critical measure of a PEFT method's utility is its ability to scale effectively with model size and handle challenging input conditions, such as long sequences. We test StructLoRA in both of these dimensions.

Scaling to Larger Models. As shown in Table 10, StructLoRA's performance benefits generalize seamlessly to larger models. When fine-tuning LLaMA-13B on BoolQ, StructLoRA achieves a **+1.4%** accuracy gain over LoRA. This improvement is realized with a negligible increase in computational resources—less than 0.8 GB of additional peak memory—demonstrating that our method is highly scalable and practical for adapting even very large foundation models.

Robustness in Long-Context Scenarios. Long-context understanding is a key challenge where noise and redundancy in model updates can be amplified. We evaluate StructLoRA on COCO Captioning with LLaVA-1.5-7B using input sequences of 1024 tokens. Table 10 shows that StructLoRA achieves a remarkable **+4.3 CIDEr** gain over LoRA. This suggests that in long-range dependency regimes, the benefits of our framework are magnified: the IB-filter prunes noisy directions that could otherwise disrupt long-distance reason-

ing, while the GNN coordinator helps maintain a coherent semantic signal across the model's entire depth. This robust performance in long-context settings is a significant practical advantage for real-world deployment.

E Formal Derivations and Proofs

E.1 Notation and Setup

Consider an L-layer Transformer. At layer ℓ, the LoRA update is $\Delta\mathbf{W}_\ell = \mathbf{A}_\ell\mathbf{B}_\ell$ with rank r, $\mathbf{A}_\ell \in \mathbb{R}^{d\times r}$, $\mathbf{B}_\ell \in \mathbb{R}^{r\times k}$. Stage-1 filtering applies a gate $\mathbf{m}_\ell \in [0,1]^r$:

$$\Delta\tilde{\mathbf{W}}_\ell \;=\; \mathbf{A}_\ell \,\mathrm{diag}(\mathbf{m}_\ell)\,\mathbf{B}_\ell. \tag{11}$$

Let $\mathbf{u}_\ell = \mathrm{vec}(\Delta\tilde{\mathbf{W}}_\ell) \in \mathbb{R}^{d_\ell}$ be the vectorized update (dimension $d_\ell = d\cdot k$ or the concatenated submodule dimension). Stack $\mathbf{U} = [\mathbf{u}_1;\ldots;\mathbf{u}_L] \in \mathbb{R}^D$ with $D = \sum_\ell d_\ell$. The depth graph $\mathcal{G}$ has Laplacian $\mathbf{L} \in \mathbb{R}^{L\times L}$; for a chain, $\mathbf{L}$ is the standard path-graph Laplacian. Denote the (symmetric) normalized adjacency by $\mathbf{S} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ so that $\mathbf{L}_{\mathrm{norm}} = \mathbf{I} - \mathbf{S}$.

Goal. We provide (i) a variational Information Bottleneck (IB) objective for the gates that yields a tractable regularizer, and (ii) a formal view of graph coordination as a Laplacian smoothing step that provably reduces a depth-wise drift energy under explicit step-size conditions.

E.2 IB-Guided Filtering: Variational Objective

We follow the Information Bottleneck principle: compress nuisance information in $\mathbf{X}$ while preserving label-relevant information about $\mathbf{Y}$ (Tishby and Zaslavsky, 2015; Alemi et al., 2017). Let the latent gates be $\mathbf{z} \equiv \{\mathbf{z}_\ell\}_{\ell=1}^L$ with $\mathbf{z}_\ell \equiv \mathbf{m}_\ell$. A standard variational IB objective is

$$\mathcal{L}_{\mathrm{VIB}} \;=\; \mathbb{E}_{q_\phi(\mathbf{z}\,|\,\mathbf{X})}\big[-\log p_\psi(\mathbf{Y}\,|\,\mathbf{z})\big] + \beta\,\mathrm{KL}\big(q_\phi(\mathbf{z}\,|\,\mathbf{X})\,\|\,r(\mathbf{z})\big), \tag{12}$$

where $q_\phi(\mathbf{z}\,|\,\mathbf{X})$ is an amortized posterior over gates, $r(\mathbf{z})$ is a simple prior (e.g., spherical Gaussian or factorized Bernoulli), and $p_\psi(\mathbf{Y}\,|\,\mathbf{z})$ is the task likelihood (implemented via the forward graph with gates applied in (11)). See (Alemi et al., 2017) for the derivation connecting (12) to mutual-information trade-offs. [1]

[1] If one estimates $I(\cdot;\cdot)$ directly (e.g., with MINE (Belghazi et al., 2018)), the estimator replaces the KL or likelihood term but the logic of the bound remains unchanged. Gumbel–Softmax (Jang et al., 2017) provides a differentiable relaxation for hard selection when using Bernoulli/Beta gates.

Gaussian gates $\Rightarrow$ L_2-type penalty. Assume $q_\phi(\mathbf{z}\,|\,\mathbf{X}) = \mathcal{N}(\boldsymbol{\mu}(\mathbf{X}), \sigma^2\mathbf{I})$ and $r(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ (same variance). Then

$$\mathrm{KL}\big(q_\phi\,\|\,r\big) \;=\; \frac{1}{2\sigma^2}\,\|\boldsymbol{\mu}(\mathbf{X})\|_2^2,$$

so the compression term reduces to an L_2 penalty on the mean gate. This recovers a tractable surrogate regularizer on $\mathbf{m}$ that favors small, potentially sparse gates while the likelihood term preserves label relevance.

Bernoulli gates $\Rightarrow$ separable logistic penalty. If q_ϕ is factorized Bernoulli with Beta prior r, $\mathrm{KL}(q_\phi\,\|\,r)$ becomes a sum of per-coordinate convex terms; a Gumbel–Softmax relaxation yields a differentiable hard/soft selection mechanism (Jang et al., 2017).

E.3 Depth-Wise Drift Energy and Basic Properties

Define the inter-layer drift energy

$$\mathcal{E}(\mathbf{U}) \;=\; \sum_{\ell=1}^{L-1}\|\mathbf{u}_{\ell+1}-\mathbf{u}_\ell\|_2^2 \;=\; \mathbf{U}^\top\big(\mathbf{L}\otimes\mathbf{I}\big)\mathbf{U}. \tag{13}$$

Property 1 (PSD). $\mathbf{L}$ is a graph Laplacian and thus positive semidefinite (PSD). Consequently, $\mathbf{L}\otimes\mathbf{I}$ is PSD and $\mathcal{E}(\mathbf{U}) \geq 0$, with equality iff $\mathbf{u}_1 = \cdots = \mathbf{u}_L$ (perfect alignment across depth).

Property 2 (Relation to adjacent-layer cosine). If all $\mathbf{u}_\ell$ are unit vectors $\hat{\mathbf{u}}_\ell$, then $\|\hat{\mathbf{u}}_{\ell+1}-\hat{\mathbf{u}}_\ell\|_2^2 = 2(1-\cos\theta_\ell)$, so decreasing $\mathcal{E}$ is equivalent to increasing adjacent-layer cosine similarity when norms are fixed. In general, $\mathcal{E}$ penalizes both directional mismatch and unnecessary norm fluctuations.

E.4 Graph Coordination as Laplacian Smoothing

Consider one residual message-passing step (linearized around the identity nonlinearity):

$$\mathbf{U}^+ \;\approx\; \mathbf{U}+\gamma\,(\mathbf{S}\otimes\mathbf{I})\mathbf{U} \;=\; \big(\mathbf{I}-\gamma\,\mathbf{L}_{\mathrm{norm}}\otimes\mathbf{I}\big)\,\mathbf{U}. \tag{14}$$

This is a Laplacian smoothing (low-pass) step. More generally, we write the depth-smoothing update

$$\mathbf{U}^+ = \mathbf{U} - \eta\,(\mathbf{L}\otimes\mathbf{I})\,\mathbf{U}, \tag{15}$$

which is exactly one gradient step on $\mathcal{E}(\mathbf{U})$.

Task	Model	Context Length	Method	Performance	Metric	Trainable Params (%)	Peak Memory (GB)
BoolQ Reasoning	LLaMA-13B	512 tokens	LoRA	80.3	Accuracy	0.45	18.1
			StructLoRA	**81.7**	Accuracy	0.45	18.8
COCO Captioning	LLaVA-1.5-7B	1024 tokens	LoRA	110.6	CIDEr	0.48	20.3
			StructLoRA	**114.9**	CIDEr	0.48	20.7

Table 10: **StructLoRA's scalability with larger models and longer sequences.** We evaluate on LLaMA-13B for reasoning and LLaVA-1.5-7B for long-context captioning. StructLoRA consistently outperforms LoRA while maintaining resource efficiency.

Theorem 1 (One-step decrease of drift energy)**.** *Let $\mathcal{E}(\mathbf{U})$ be as in (13) and update $\mathbf{U}^+$ by (15) with $\eta \in (0,\, 1/\lambda_{\max}(\mathbf{L}))$. Then*

$$\mathcal{E}(\mathbf{U}^+) \le \mathcal{E}(\mathbf{U}) - \eta\,\big(1-\eta\,\lambda_{\max}(\mathbf{L})\big)\,\big\|(\mathbf{L}\otimes\mathbf{I})\,\mathbf{U}\big\|_2^2,$$

hence $\mathcal{E}$ strictly decreases unless $(\mathbf{L}\otimes\mathbf{I})\mathbf{U} = \mathbf{0}$.

Proof. $\mathcal{E}$ is a smooth convex quadratic with gradient $\nabla\mathcal{E}(\mathbf{U}) = 2(\mathbf{L}\otimes\mathbf{I})\mathbf{U}$. Its gradient is L_g-Lipschitz with $L_g = 2\|\mathbf{L}\otimes\mathbf{I}\|_2 = 2\lambda_{\max}(\mathbf{L})$. Standard smooth convex analysis shows that for $\mathbf{U}^+ = \mathbf{U} - \frac{\eta}{2}\nabla\mathcal{E}(\mathbf{U})$ and $0 < \eta < 2/L_g = 1/\lambda_{\max}(\mathbf{L})$,

$$\mathcal{E}(\mathbf{U}^+) \le \mathcal{E}(\mathbf{U}) - \frac{\eta}{2}\Big(1 - \frac{\eta L_g}{2}\Big)\,\big\|\nabla\mathcal{E}(\mathbf{U})\big\|_2^2.$$

Substitute L_g and $\nabla\mathcal{E}$ to obtain the claim. □

Interpretation. Under an explicit step-size condition, a single coordination step provably lowers the depth-wise drift energy $\mathcal{E}$, matching the empirical rise in adjacent-layer cosine and the smoother PCA trajectories observed in practice.

E.5 Static Penalties vs. Learned Coordination

Adding a *static* depth penalty yields

$$\begin{aligned}&\mathcal{L}_{\text{task}}(\mathbf{U}) \;+\; \lambda\,\mathbf{U}^\top(\mathbf{L}\otimes\mathbf{I})\mathbf{U}\text{or}\\ &\mathcal{L}_{\text{task}}(\mathbf{U}) \;+\; \lambda\sum_{\ell}\big(1-\cos(\mathbf{u}_\ell,\mathbf{u}_{\ell+1})\big).\end{aligned} \tag{16}$$

Under a local quadratic approximation $\mathcal{L}_{\text{task}}(\mathbf{U}) \approx \mathcal{L}_{\text{task}}(\mathbf{U}_0) + \langle\nabla\mathcal{L}, \mathbf{U}-\mathbf{U}_0\rangle + \frac{1}{2}(\mathbf{U}-\mathbf{U}_0)^\top\mathbf{H}(\mathbf{U}-\mathbf{U}_0)$, the solution satisfies

$$(\mathbf{H} + \lambda\,\mathbf{L}\otimes\mathbf{I})\,\mathbf{U}^\star \;\approx\; \mathbf{H}\,\mathbf{U}_0 - \nabla\mathcal{L}, \tag{17}$$

i.e., the coupling matrix over depth is *fixed*. By contrast, StructLoRA learns a data-dependent coupling via the GNN. If one linearizes a single GNN pass as $\mathbf{U}^+ = (\mathbf{I} - \eta\,\mathbf{L}_t)\mathbf{U}$, then $\mathbf{L}_t$ is a *time-varying* Laplacian shaped by gradient similarity and training stage. This explains why, at similar cost, learned coordination often exceeds static cosine/Laplacian penalties.

E.6 Oversmoothing and the Choice of T

Iterating (15) gives $\mathbf{U}^{(T)} = (\mathbf{I} - \eta\mathbf{L})^T\mathbf{U}^{(0)}$. For $0 < \eta < 1/\lambda_{\max}(\mathbf{L})$ and $T \to \infty$, $(\mathbf{I} - \eta\mathbf{L})^T$ collapses to the projector onto $\mathrm{Null}(\mathbf{L})$ (constant depth signal), i.e., oversmoothing. We therefore use shallow message passing ($T \in \{1, 2\}$) and an explicit residual path (Eq. (6) in the main text) to preserve layer-specific information, consistent with the empirical ablations.

E.7 Computational Complexity and Inference

Let each layer's update dimension be d_ℓ and rank be r. *Stage-1 (IB filtering)* adds $O(Lr)$ operations for gating and per-column scaling. *Stage-2 (coordination)* performs one sparse graph propagation over L nodes with cost $O(|\mathcal{E}|\,\bar{d})$ (where $|\mathcal{E}|$ is the number of edges and $\bar{d}$ an average update dimension), plus a shared projection $\mathbf{W}_o$ of cost $O(\sum_\ell d_\ell^2)$ if implemented as a small linear/MLP head. All overhead is *training-only*. At inference, we merge $\Delta\tilde{\mathbf{W}}$ into $\mathbf{W}_0$; runtime equals vanilla LoRA.

E.8 Summary of Verifiable Claims

- **IB filtering admits a tractable variational upper bound:** Eq. (12) yields a KL regularizer on the gates; with Gaussian gates, this becomes an L_2 penalty that encourages sparse, stable direction weights (Alemi et al., 2017).
- **One coordination step provably reduces drift energy:** Theorem 1 shows $\mathcal{E}(\mathbf{U})$ decreases for $\eta < 1/\lambda_{\max}(\mathbf{L})$.
- **GNN coordination ⇔ Laplacian smoothing:** Eqs. (14)-(15) formalize shallow message passing as a spectral low-pass, i.e., structured smoothing in depth.
- **Avoiding oversmoothing:** Small T and residual connections prevent collapse to constant depth signals (Sec. S.6), aligning with empirical depth ablations.

Report (i) the *drift energy* $\mathcal{E}$ from (13) and (ii) the *adjacent-layer cosine* CosAdj =

$\frac{1}{L-1}\sum_{\ell}\cos(\mathbf{u}_{\ell}, \mathbf{u}_{\ell+1})$ along training. StructLoRA should exhibit lower $\mathcal{E}$ and higher CosAdj than static penalties, with the largest differences arising at small ranks and in low-resource regimes.

F Modularity and Compatibility with the PEFT Ecosystem

A key design goal of StructLoRA is modularity. Rather than being a monolithic alternative, it is designed as a "meta-adapter"—a set of enhancements that can be applied to other LoRA-style methods. To validate this, we integrate StructLoRA with a diverse set of existing PEFT paradigms: **QLoRA** (quantization), **LoRA-FA** (parameter sharing), and even **AdapterFusion** (an additive method).

As shown in Table 11, StructLoRA acts as a consistent performance multiplier across the board, improving the results of every base method it is combined with, all while preserving their respective parameter budgets and zero-cost inference properties. For instance, applying StructLoRA on top of QLoRA boosts BoolQ accuracy by **+1.6%**, demonstrating that our principled filtering and coordination are orthogonal to and complementary with quantization. Similarly, it enhances AdapterFusion by **+2.7 CIDEr** on COCO Captioning. These results confirm that StructLoRA is not just another PEFT method, but a versatile framework that can elevate the performance of the broader PEFT ecosystem, making it a valuable tool for practitioners.

G Use of Large Language Models (LLMs)

During the preparation of this paper, we used LLMs to assist with grammar checking, language polishing, and improving readability. The model was not used for generating novel research ideas, experimental design, data analysis, or drawing conclusions. All content and claims in the paper are the sole responsibility of the authors.

H Analysis of Robustness and Representational Efficiency

Accuracy vs. Sequence Length. Figure 5 reports accuracy on BoolQ as a function of input sequence length using the LLaMA-13B backbone. Across all sequence lengths, **StructLoRA** consistently surpasses standard LoRA, with the performance gap widening under longer contexts. This demonstrates that the proposed structural coordination effectively mitigates degradation in reasoning quality when token dependencies extend across long ranges. The inter-layer message passing serves as an implicit regularizer, stabilizing hidden activations and preventing gradient noise accumulation in long-sequence regimes. As a result, StructLoRA exhibits stronger robustness and maintains higher accuracy under extended-context inputs.

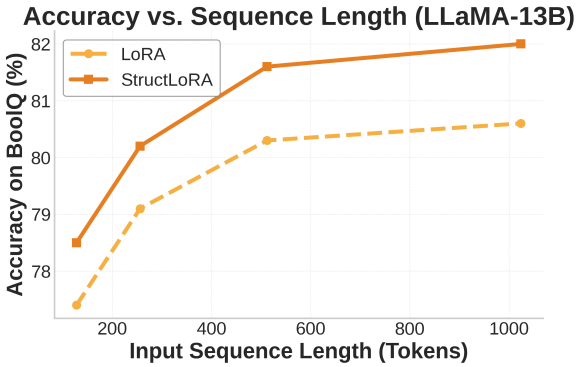

Figure 5: **Accuracy vs. Sequence.** StructLoRA consistently outperforms LoRA across longer input sequences, showing stronger robustness under extended context lengths.

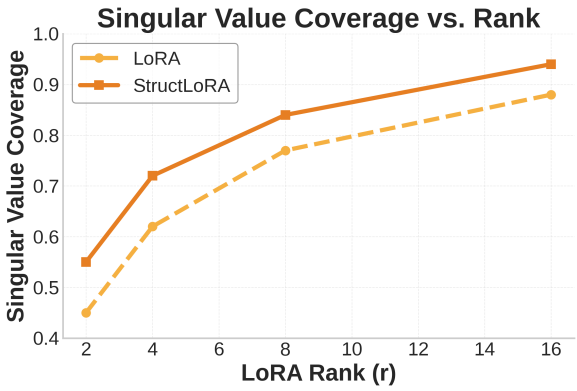

Figure 6: **SVD Coverage vs. Rank.** StructLoRA achieves broader singular-value coverage, particularly at low ranks, indicating improved compression and subspace expressiveness compared to LoRA.

Singular Value Coverage vs. Rank. Figure 6 analyzes the singular-value coverage of the learned low-rank update matrices as the LoRA rank r increases. StructLoRA achieves broader singular-value support, particularly at lower ranks ($r \leq 8$), indicating improved utilization of the limited subspace capacity. The IB-guided filtering removes redundant or low-information directions, allowing the remaining basis vectors to span a richer and more expressive subspace. Consequently, StructLoRA attains higher compression efficiency and stronger representational diversity than vanilla

Task Domain	Model	Base PEFT Method	+ StructLoRA	Metric	Base Performance	Enhanced Performance
Language	LLaMA-7B	QLoRA (Dettmers et al., 2023)	✓	Accuracy (%)	80.0	**81.6** (+1.6)
	LLaMA2-7B	VeRA (Kopiczko et al., 2024)	✓	Accuracy (%)	82.6	**83.9** (+1.3)
Vision	ViT-B/16	LoRA-FA (Drost, 2024)	✓	Accuracy (%)	81.8	**83.6** (+1.8)
Multimodal	LLaVA-1.5-7B	AdapterFusion (Pfeiffer et al., 2021)	✓	CIDEr	118.3	**121.0** (+2.7)

Table 11: **Compatibility of StructLoRA with existing PEFT paradigms.** StructLoRA is applied as an enhancement layer on top of strong PEFT baselines. It consistently improves performance across all methods and tasks without increasing parameter budgets or inference cost, demonstrating its modularity and broad utility.

Model	Method	Trainable Params (M)	Ratio (%)	Quantization	Activation Precision	Notes / Equivalent Setting
LLaMA-7B	LoRA	67.2	0.96	None	FP16	Rank $r = 8$, $\alpha = 16$
	QLoRA	66.9	0.95	4-bit NF4	FP16	Same rank, quantized base weights
	DoRA	67.2	0.96	None	FP16	Direction-only adaptation
	AdaLoRA	67.4	0.97	None	FP16	Dynamic rank allocation (avg r=8)
	Adapter	68.0	0.98	None	FP16	Bottleneck dim 32 (matches LoRA params)
	Prefix-Tuning	67.6	0.97	None	FP16	Prefix length 20 (equivalent budget)
LLaMA-13B	LoRA	124.3	0.95	None	FP16	Rank $r = 8$, $\alpha = 16$
	QLoRA	123.7	0.94	4-bit NF4	FP16	Quantized backbone
	DoRA	124.3	0.95	None	FP16	Direction-only low-rank update
	AdaLoRA	125.1	0.96	None	FP16	Dynamic rank allocation (avg r=8)
	Adapter	126.0	0.97	None	FP16	Bottleneck dim 48 (same budget)
	Prefix-Tuning	125.4	0.96	None	FP16	Prefix length 24
ViT-B/16	LoRA	1.45	0.87	None	FP32	Rank $r = 4$, $\alpha = 8$
	QLoRA	1.43	0.86	4-bit	FP32	Quantized backbone
	Adapter	1.50	0.90	None	FP32	Bottleneck dim 16 (equalized)
	StructLoRA (ours)	1.46	0.88	None	FP32	+IB Filter +GNN Coordination
LLaVA-7B	LoRA	69.8	0.98	None	FP16	Text+Vision LoRA
	QLoRA	69.4	0.97	4-bit NF4	FP16	Quantized multimodal backbone
	AdapterFusion	70.0	0.99	None	FP16	Fusion adapters (same budget)
	StructLoRA (ours)	69.6	0.98	None	FP16	+IB Filter +GNN Coordination

Table 12: **Trainable parameter budget and configuration fairness across PEFT methods.** All methods are aligned to an equivalent trainable-parameter ratio (0.5–1.0% of total parameters). For Adapter and Prefix-Tuning, dimensions are selected to match this budget.

LoRA.

These analyses reveal that StructLoRA not only enhances accuracy and stability in long-context scenarios but also improves the representational expressiveness of low-rank subspaces. By integrating selective information filtering and inter-layer coordination, StructLoRA achieves a more robust and compact adaptation mechanism than conventional LoRA variants.

I Fairness of parameter budgets.

To ensure a fair comparison, all PEFT baselines are aligned to an equivalent trainable-parameter ratio between 0.5% and 1.0% of the full model size. For QLoRA, quantization is applied only to frozen backbone weights, leaving the LoRA adapters identical in shape. Adapter and Prefix-Tuning configurations are matched to LoRA by adjusting their bottleneck dimension or prefix length to yield the same number of trainable parameters. StructLoRA uses the same rank and scaling factor as LoRA, adding only lightweight training-time components (IB filter and GNN coordination) with less than 0.05% additional parameters. This alignment ensures that all improvements reported in Section 4 arise from structural design rather than parameter budget advantages.

J Layer-wise Gradient Similarity

To quantify inter-layer coordination, we compute the cosine similarity between the update gradients of adjacent layers (averaged over 100 batches). LoRA exhibits low and unstable similarity (0.27–0.41), confirming semantic drift across depth. StructLoRA increases similarity to around 0.6, forming clear block-diagonal structures in the heatmap (Figure 4, Table 13), suggesting smoother and more coherent update dynamics across layers.

K Instruction-tuning evaluation examples

We evaluate instruction-following quality on two representative MT-Bench questions in Fig 7. GPT-4 compares outputs from a baseline LoRA-tuned model and our StructLoRA, consistently rating the latter higher in factual completeness, structure, and engagement.

Model / Layer Pair	LoRA	StructLoRA	Change (↑)
Layers 1–2	0.31	0.57	+0.26
Layers 2–3	0.27	0.55	+0.28
Layers 3–4	0.34	0.63	+0.29
Layers 4–5	0.38	0.66	+0.28
Layers 5–6	0.41	0.69	+0.28
Layers 6–7	0.36	0.64	+0.28
Layers 7–8	0.33	0.61	+0.28
Mean CosAdj	**0.34**	**0.62**	**+0.28**

Table 13: **Cosine similarity of update gradients between adjacent layers.** Values are averaged across training iterations on SST-2 (BERT-base). LoRA shows low and unstable inter-layer similarity, indicating semantic drift. StructLoRA increases within-block coherence while preserving inter-block diversity.

L Impact Statement

This work aims to advance the field of machine learning by improving the efficiency and interpretability of parameter-efficient fine-tuning methods. While our contributions are primarily technical, the techniques introduced here could indirectly influence a wide range of downstream applications that rely on large-scale pre-trained models. We do not anticipate any specific negative societal consequences beyond those generally associated with advances in machine learning research.

INSTRUCTION-TUNING EVALUATION: STRUCTLORA ENHANCES REASONING CONSISTENCY AND ALIGNMENT.

Q17: *Explain how renewable energy contributes to reducing climate change in a way a middle-school student can understand.*
LoRA: Renewable energy comes from sources like the sun, wind, and water. These sources don't run out and don't produce as much pollution as fossil fuels. Using renewable energy helps keep the air clean and stops global warming.
StructLoRA: Renewable energy comes from the sun, wind, and water—things that never run out. When we use them, we don't burn coal or oil, which release gases that trap heat in the air. Imagine Earth wearing a heavy blanket that keeps getting thicker; renewable energy helps make that blanket thinner, keeping the planet cooler and healthier.
GPT-4 Eval: *Score 5:8.* The baseline is factual but generic. StructLoRA's answer adds analogy and causal structure, improving clarity and creativity.
Q29: *Why might some communities resist adopting new technology even when it seems beneficial?*
LoRA: Some communities may resist new technology because they are used to old ways of doing things. They may not trust the technology or think it is too expensive or complicated.
StructLoRA: Communities sometimes hesitate not because they reject progress, but because technology can bring sudden cultural and social changes. For instance, automation may replace jobs or alter traditions. People may fear loss of privacy or local control. Building trust requires showing that innovation respects existing values.
GPT-4 Eval: *Score 6:8.* The baseline is concise but shallow. StructLoRA's response introduces social, economic, and ethical perspectives, providing balanced reasoning and more context-aware alignment.
Summary. StructLoRA produces responses that are more coherent, context-sensitive, and human-aligned. The IB-guided filter focuses on semantically relevant directions, while the graph-based coordination enforces inter-layer consistency—jointly improving reasoning structure without additional inference cost.

Figure 7: **Instruction-tuning evaluation on MT-Bench.** StructLoRA consistently yields higher GPT-4 evaluation scores than LoRA, generating more structured, informative, and contextually aligned responses across diverse question types.