PRISM: SPECTRAL ANALYSIS AIDED NESTED MANIFOLD LEARNING FOR EFFICIENT MULTIMODAL LARGE LANGUAGE MODEL ADAPTATION ON EDGE

Anonymous authors
Paper under double-blind review

ABSTRACT

Parameter-efficient fine-tuning (PEFT) has established itself as a dominant paradigm for adapting lightweight multimodal large language models, aiming to balance adaptation efficacy with computational efficiency. However, prevalent approaches predominately adhere to static parameterization, imposing a monolithic computational regimen that is insensitive to the intrinsic variance in visual complexity. This rigidity induces significant computational redundancy, thereby hindering optimal inference efficiency, particularly for adaptation on the edge. To surmount this bottleneck, we introduce a dynamic PEFT framework, dubbed probabilistic representation via information-theoretic spectral modeling (PRISM). Synthesizing spectral analysis with nested manifold learning, PRISM exploits frequency-domain features to identify the intrinsic low-dimensional manifold of task-specific parameters. This mechanism aids the precise, on-demand alignment between computational resources and instance-level complexity by dynamically traversing a nested parameter subspace. Extensive empirical validation demonstrates that, even utilizing a mere 2B-parameter model, PRISM maintains high accuracy, surpassing the Gemini 3 Pro in specialized domains. Furthermore, compared to mainstream PEFT methodologies, PRISM exhibits superior performance in terms of both accuracy and inference latency.

1 INTRODUCTION

In recent years, multimodal large language models (MLLMs) have demonstrated remarkable capabilities in understanding interactive visual scenes, performing complex reasoning, and generating natural language descriptions Zhang et al. (2024). However, as model parameters scale from billions to hundreds of billions, the demand for computational resources, memory bandwidth, and power consumption has grown exponentially. This creates a significant disparity between resource requirements and the strict physical constraints of edge devices Liu et al. (2024). While cloud-based inference offers powerful computational support, deploying MLLMs directly to edge devices has become a critical path for next-generation ubiquitous artificial intelligence (AI), primarily due to concerns regarding privacy, network latency, and bandwidth costs.

Currently, the core conflict in edge deployment is not merely the model scale. Rather, it is the fundamental mismatch between the prevailing static inference paradigm—which utilizes full parameters and all tokens—and the dynamic complexity of input data Han et al. (2021). In real-world edge perception scenarios, visual inputs exhibit high information variance. For instance, the computational cost required to process an image of an object with a simple background should be significantly lower than that required for parsing complex medical imagery or crowded street views He et al. (2022).

Existing lightweight adaptation methods for multimodal large models have not effectively resolved this contradiction. Mainstream parameter-efficient fine-tuning (PEFT) techniques, such as low-rank adaptation (LoRA) and its variants Hu et al. (2022); Zhang et al. (2023); yang Liu et al. (2024), successfully reduce storage overhead through low-rank matrices. However, they do not account for the input itself. Regardless of the difficulty of the input sample, these models perform inference within a fixed rank or parameter subspace. This static nature leads to significant computational

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.

redundancy for simple samples, while complex samples are constrained by a fixed parameter capacity. Consequently, it becomes difficult to balance inference efficiency with representation performance.

On the other hand, the research community has explored gating mechanisms Shazeer et al. (2017); Fedus et al. (2022); Dai et al. (2024) and visual sparsification strategies Rao et al. (2021); Yin et al. (2022); Ye et al. (2025) to achieve dynamic inference. Yet, most existing methods rely on coarse-grained routing strategies or heuristic token pruning Bolya et al. (2023); Li et al. (2025). They lack a deep understanding of the parameter manifold structure. Furthermore, existing dynamic inference schemes often struggle to precisely perceive the intrinsic heterogeneity of multimodal inputs. This results in an inability to establish an effective mapping between instance-level complexity and computational resources.

To overcome this bottleneck, we posit that the adaptation of edge-side multimodal models must be re-examined through the lens of spectral analysis. We argue that visual tasks of varying complexities manifest as distinct spectral distributions in the feature space: low-frequency signals can be effectively captured by low-dimensional parameter manifolds, whereas high-frequency signals require higher-dimensional parameter subspaces. Based on this insight, we propose a dynamic PEFT framework named PRISM (Probabilistic Representation via Information-Theoretic Spectral Modeling).

Distinct from traditional static adaptation or simple mixture of experts (MoE) routing, PRISM introduces a Nested Manifold Learning mechanism. It utilizes frequency domain features of the input data as spectral fingerprints to dynamically drive the model's traversal within predefined nested parameter subspaces, without requiring repetitive training. Essentially, this mechanism constructs a Spectral-driven MoE. It enables the model to activate parameter manifolds of different dimensions on demand, based on the intrinsic information entropy of the input. This achieves an optimal trade-off between computational efficiency and inference accuracy in strictly constrained edge environments.

2 Related Works

PEFT aims to adapt large pre-trained models by training only a small number of parameters and has become the de facto standard for edge deployment. LoRA Hu et al. (2022) established the foundation of this field by introducing low-rank adaptation matrices. Subsequent research has focused on micro-optimizations in storage and training efficiency. For example, AdaLoRA Zhang et al. (2023) addresses the parameter budget allocation problem, QLoRA Dettmers et al. (2023) combines quantization techniques to lower the memory threshold, and LoftQ Li et al. (2024) further optimizes the quantization initialization process.

To achieve extreme compression rates, VeRA Kopiczko et al. (2024) utilizes frozen random orthogonal matrices to further reduce trainable parameters. Conversely, to address the limited learning capacity of LoRA, DoRA yang Liu et al. (2024) improves stability via weight decomposition, PiSSA Meng et al. (2024) utilizes singular value decomposition (SVD) principal component initialization to accelerate convergence, and FlyLoRA Zou et al. (2025) attempts to resolve parameter interference through the orthogonality of random matrices. Although these methods have made significant progress in storage efficiency, they inherently impose a single, rigid computational pattern. Whether it is the rank allocated in AdaLoRA or the matrices frozen in VeRA, the computation graph is fixed once training is complete. This static nature during inference prevents the model from handling dynamic loads at the edge, leading to computational redundancy for simple samples and limited capability for complex ones.

To break the constraints of static computation, dynamic neural networks introduce conditional computation mechanisms. As a foundational work for gating networks and MoE, Shazeer et al. (2017) proposed activating expert modules on demand via Gating. The Switch Transformer Fedus et al. (2022) subsequently achieved stable training of large-scale sparse models by simplifying the gating mechanism, while Dai et al. (2024) further proposed fine-grained expert segmentation.

In the visual domain, researchers have attempted to introduce this dynamism at the architecture level. For instance, the dynamic token sparsification framework proposed in Rao et al. (2021), the adaptive computation strategy in Yin et al. (2022), and the token merging and pruning techniques in Bolya et al. (2023) and Ye et al. (2025) all aim to reduce redundant tokens during inference. However, these methods primarily save computation by "skipping" certain layers, modules, or tokens. They often rely on heuristic routing strategies and ignore the intrinsic low-dimensional structure of the

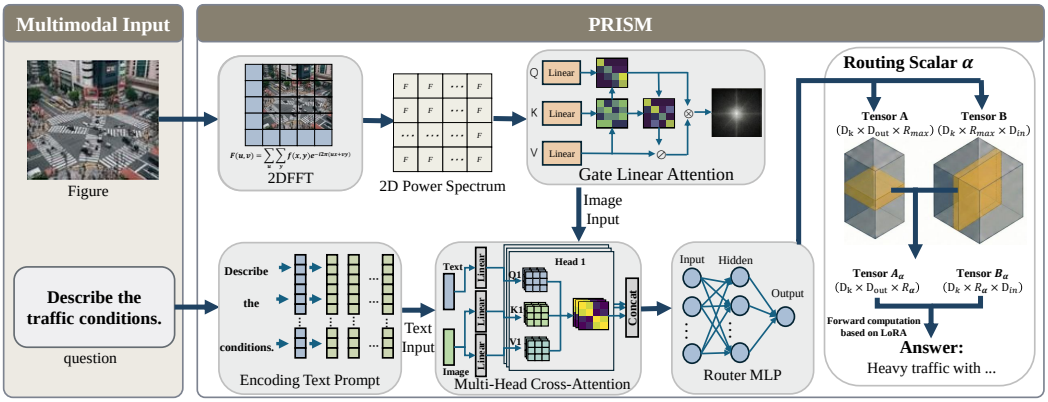


Figure 1: Overview of PRISM. Given a multimodal input, PRISM estimates task-conditioned spectral complexity and dynamically activates different ranks within a nested LoRA manifold.

parameter space. While this coarse-grained dynamism improves throughput, the lack of fine-grained perception of the parameter manifold topology often leads to a degradation in generalization ability under constrained resources.

Efficient adaptation of multimodal models is essentially a manifold alignment problem: how to map high-dimensional visual features into the semantic space of language models. Radford et al. (2021) and Jia et al. (2022) pioneered the use of semantic prompts and text description matching to enhance the transfer ability of visual models, attempting to find commonalities at the feature level. Subsequent works, such as the semantic contrastive learning paradigm introduced in Liang et al. (2022), the text-semantic guided adapter generation in Karmanov et al. (2024), and the improved visual projector designs in Cha et al. (2024) and Li et al. (2025) for preserving local context, all essentially aim to capture the semantic manifold of input data more precisely. However, these efforts are mainly confined to representation learning at the input or feature level. They fail to propagate this perception of semantic complexity down to the process of model weight generation. Moreover, these methods typically operate in the spatial domain, where the intrinsic dimensionality of features is often entangled and difficult to quantify, whereas our spectral approach offers a disentangled view of information density.

To address the blindness of static adaptation to the intrinsic spectral heterogeneity of visual tasks, we propose PRISM, a dynamic framework grounded in the physical insight that low-frequency signals occupy low-dimensional manifolds while high-frequency textures necessitate high-rank reconstruction. By establishing the first theoretical link between spectral entropy and parameter manifold dimensionality, PRISM integrates Nested Manifold Learning with a training-free spectral router to enable on-demand subspace activation. This spectral-driven Mixture-of-Experts approach not only constructs a robust physical proxy for dynamic rank selection but also achieves state-of-the-art performance with a compact 2B backbone—surpassing full-parameter models in specialized domains like remote sensing and reducing inference latency by 11.6% compared to static high-rank baselines, thereby securing an optimal efficiency-accuracy trade-off for edge deployment.

3 METHODOLOGY

PRISM establishes an instance-aware dynamic adaptation mechanism by modeling the parameter generation as a condition-driven sequence:

$$(v, t) \xrightarrow{\text{Estimation}} \mathcal{H}(v|t) \xrightarrow{\text{Routing}} R_\alpha \xrightarrow{\text{Slicing}} \Delta W_{R_\alpha}, \tag{1}$$

where v and t denote the visual and textual inputs, respectively. The multimodal input first induces a complexity measure via spectral entropy $\mathcal{H}(v|t)$, which is subsequently mapped to an adaptive intrinsic dimension R_α, and finally instantiated as a low-rank parameter update ΔW_{R_α}. Unlike conventional LoRA with a fixed capacity, PRISM dynamically expands or shrinks its active parameter manifold based on the specific complexity required by the current task.

3.1 PROBLEM FORMULATION

Let $W_{pre} \in \mathbb{R}^{D_{out} \times D_{in}}$ denote the pre-trained MLLM parameters residing on a parameter manifold $\mathcal{W}$. For a given multimodal instance $\mathcal{I} = (v, t)$, parameter-efficient adaptation seeks an update matrix $\Delta W \in T_{W_{pre}}\mathcal{W}$ in the local tangent space such that the empirical risk $\mathcal{L}$ is minimized:

$$\min_{\Delta W} \mathbb{E}_{\mathcal{I} \sim \mathcal{D}} \left[\mathcal{L}\big(W_{pre} + \Delta W(\mathcal{I})\big) \right]. \tag{2}$$

Conventional LoRA constrains ΔW to a single static sub-manifold of fixed rank r. In contrast, PRISM hypothesizes that diverse multimodal tasks require varying degrees of freedom. Therefore, we define a nested hierarchy of low-rank adaptation sub-manifolds:

$$\mathcal{S}_1 \subset \mathcal{S}_2 \subset \cdots \subset \mathcal{S}_{R_{max}} \subset T_{W_{pre}}\mathcal{W}, \tag{3}$$

where R_{max} is the predefined maximum adaptation capacity.

Instead of uniformly applying R_{max}, PRISM formulates dynamic adaptation as finding the minimal sufficient sub-manifold for each instance. We introduce an instance-specific effective rank $d_{eff}(\mathcal{I})$ and learn a dynamic parameterization function $\Phi(\cdot)$:

$$\Phi : \mathcal{I} \mapsto \Delta W_{\mathcal{I}} \in \mathcal{S}_{r^*}, \quad \text{s.t.} \quad r^* = \lceil d_{eff}(\mathcal{I}) \rceil \leq R_{max}. \tag{4}$$

This formulation ensures that the adaptation capacity is explicitly bounded by the intrinsic complexity of the input, optimizing both expressivity and computational efficiency.

3.2 SPECTRAL COMPLEXITY ESTIMATION

We estimate task complexity in the frequency domain, where structural information is more explicitly reflected through spectral distributions.

Given a visual feature map $X \in \mathbb{R}^{H \times W \times C}$, we compute its 2D Fourier spectrum:

$$P(f_x, f_y) = \left\| \sum_{i,j} X(i,j) e^{-i2\pi(f_x i + f_y j)} \right\|_2^2. \tag{5}$$

The resulting spectrum is further partitioned into non-overlapping frequency patches and linearly projected into spectral tokens, yielding $S \in \mathbb{R}^{L \times d_s}$, where L denotes the number of spectral tokens and d_s is the embedding dimension.

To suppress noisy high-frequency components while preserving semantic structures, we introduce Gate Linear Attention (GLA):

$$Q_g = SW_Q^{(gla)}, \quad K_g = SW_K^{(gla)}, \quad V_g = SW_V^{(gla)}, \tag{6}$$

$$G = \text{Sigmoid}(\text{RMSNorm}(Q_g) \odot K_g), \tag{7}$$

where $\odot$ denotes element-wise feature interaction rather than token-wise attention.

$$Z_{spec} = (V_g \odot G)W_O^{(gla)} + S, \tag{8}$$

where $W_Q^{(gla)}, W_K^{(gla)}, W_V^{(gla)}$, and the output projection $W_O^{(gla)}$ are task-independent learnable weight matrices. The resulting representation Z_{spec} forms a denoised spectral embedding for downstream complexity estimation.

3.3 TASK-DRIVEN SPECTRAL ROUTING

Since visual complexity is task-dependent, PRISM performs task-conditioned spectral querying through cross-attention.

Let text embeddings be $E_{text} \in \mathbb{R}^{M \times d}$, where M denotes the text sequence length and d is the hidden dimension. The task query and spectral key/value are computed using a distinct set of projection matrices:

$$Q = E_{text}W_Q^{(task)}, \quad K = Z_{spec}W_K^{(task)}, \quad V = Z_{spec}W_V^{(task)}. \tag{9}$$

The task-aware spectral representation is:

$$A_{score} = \frac{QK^T}{\sqrt{d}}, \quad O_{task} = \text{Softmax}(A_{score})V. \tag{10}$$

We define the **Task-Conditional Spectral Entropy**:

$$\mathcal{H}(v|t) = -\sum_k \hat{w}_k \log \hat{w}_k, \tag{11}$$

$$\hat{w}_k = \frac{1}{M}\sum_{i=1}^{M}[\text{Softmax}(A_{score})]_{i,k}. \tag{12}$$

Low entropy indicates concentrated spectral dependency and requires only compact adaptation subspaces, whereas high entropy implies broader spectral requirements and larger adaptation capacity. Intuitively, broader spectral distributions require higher-dimensional adaptation subspaces to preserve diverse structural responses.

A lightweight Router MLP maps the entropy into a routing coefficient:

$$\alpha = f_{router}(\mathcal{H}(v|t)), \tag{13}$$

where $\alpha \in [0, 1]$ represents the normalized intrinsic manifold dimension.

3.4 Nested Manifold Slicing

To efficiently instantiate variable-rank adapters, we construct shared hyper-tensors:

$$\mathcal{T}_A \in \mathbb{R}^{K\times D_{out}\times R_{max}}, \quad \mathcal{T}_B \in \mathbb{R}^{K\times R_{max}\times D_{in}}, \tag{14}$$

where K denotes the number of transformer layers equipped with adaptive LoRA modules.

The routing coefficient determines the activated rank:

$$R_\alpha = \lceil \alpha R_{max} \rceil. \tag{15}$$

The adapter weights are dynamically sliced as:

$$A_\alpha = \mathcal{T}_A[:, :, 0 : R_\alpha], \quad B_\alpha = \mathcal{T}_B[:, 0 : R_\alpha, :]. \tag{16}$$

The instantiated low-rank adaptation is therefore defined as:

$$\Delta W_{R_\alpha} = B_\alpha A_\alpha. \tag{17}$$

This defines nested subspaces:

$$\mathcal{S}_1 \subset \cdots \subset \mathcal{S}_{R_{max}}, \tag{18}$$

allowing PRISM to select the smallest manifold sufficient for the task.

Assuming the task-relevant spectral energy follows an exponentially concentrated distribution, the approximation error of nested low-rank slices satisfies the following bound.

Theorem 3.1 (Spectral Concentration Bound)**.** *For a task with spectral entropy $\mathcal{H}(v|t)$, the approximation error of a rank-r slice satisfies:*

$$\mathcal{E}(r) \le C \exp\left(-\gamma \frac{r}{\mathcal{H}(v|t)}\right), \tag{19}$$

where C, γ are constants.

Proof is deferred to Appendix A.1.

To stabilize training across dynamically selected ranks, we derive an Iso-stability Scaling Law:

$$\beta(R_\alpha) \propto \frac{1}{\sqrt{R_\alpha}}. \tag{20}$$

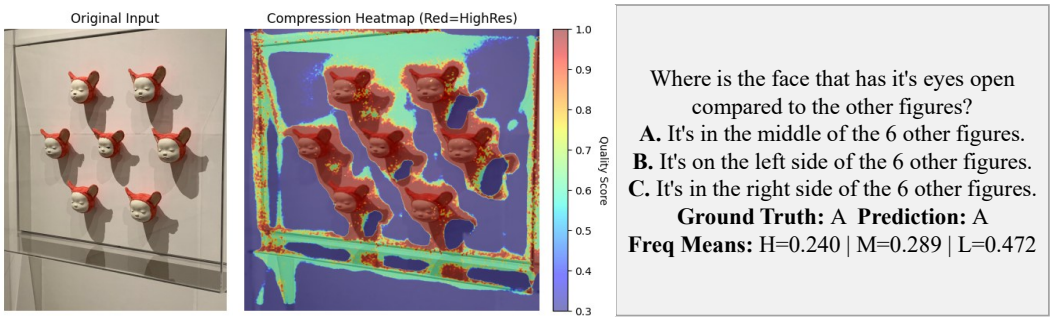


Figure 2: Visualization of the spectral complexity landscape used by PRISM.

The final adaptation rule becomes:

$$h_{out} = W_{pre}h_{in} + \frac{\lambda_0}{\sqrt{R_\alpha}}(B_\alpha A_\alpha)h_{in}, \tag{21}$$

where h_{in} and h_{out} denote the input and output hidden states of the adapted layer, respectively, and λ_0 is a constant base scaling hyperparameter.

Consequently, PRISM forms a complete adaptive chain:

$$(v, t) \to \mathcal{H}(v|t) \to R_\alpha \to \Delta W_{R_\alpha}, \tag{22}$$

enabling dynamic low-rank adaptation conditioned on task complexity.

4 Experiments

This chapter provides a comprehensive evaluation of the proposed PRISM framework. We integrate PRISM's unified spectral-guided nested learning mechanism into the Qwen3-VL-2B backbone. The experiments aim to achieve three goals. First, we validate the accuracy improvements across general, basic, and specialized visual tasks. Second, we assess the inference latency to ensure the feasibility of manifold-based dynamic adaptation on edge devices. Third, we verify the necessity of the spectral analysis mechanism through ablation studies. The discussion follows a logical sequence, progressing from experimental settings to the validation of the spectral analysis mechanism and main experimental results, followed by ablation studies and hyperparameter analysis of the nested manifold structure.

4.1 Experimental Settings

We adopt Qwen3-VL-2B as the foundational backbone. Characterized by its lightweight architecture and multimodal capabilities, this model serves as an optimal proxy for evaluating deployment viability in resource-constrained environments. The experimental design is structured to validate PRISM as a holistic framework that dynamically modulates model capacity in response to input complexity, thereby establishing an optimal trade-off between adaptation performance and computational efficiency.

To rigorously assess this dynamic mechanism across a broad spectrum of visual manifolds, we curate a comprehensive benchmark suite comprising four distinct datasets. We utilize SimpleVQA (Cheng et al., 2025) as a baseline to represent elemental visual concepts with low intrinsic dimensionality, whereas RealworldQA challenges the model's robustness within complex, unconstrained open-world environments. To evaluate higher-order cognitive capabilities, we incorporate MMStar (Chen et al., 2024), which targets advanced multimodal reasoning and logical deduction. Finally, XLRS-Bench (Wang et al., 2025) is employed to verify domain generalization within remote sensing, representing a specialized high-frequency domain characterized by dense spatial details.

To simulate the resource constraints typical of edge devices, we downsample visual inputs to a resolution of 504×504 (please refer to Appendix for the detailed rationale), excluding the XLRS-Bench dataset. We benchmark PRISM against mainstream PEFT methods such as DoRA and

Table 1: Quantitative comparison on RealworldQA, SimpleVQA, XLRS-Bench, and MMStar using the **Qwen3-VL-2B** backbone. Results are averaged over three runs (seed=42). Metrics include Accuracy (Acc, ↑) and inference latency (Lat, ↓) on an NVIDIA RTX 5090 GPU. Avg Acc denotes the average across all benchmarks. Values in parentheses indicate performance gains over Base-Qwen. Bold indicates the best result.

Method	**RealworldQA** Acc (↑)	**SimpleVQA** Acc (↑)	**MMStar** Acc (↑)	**XLRS-Bench** Acc (↑)	**Avg Acc** Acc (↑)	**Avg Latency** (s) (↓)
Qwen3-Vl-2B	63.94%	32.61%	53.33%	33.12%	45.75%	–
Gemini3Pro	**73.96%** (+10.02%)	**41.63%** (+9.02%)	**61.33%** (+8.00%)	46.43% (+13.31%)	**55.84%** (+10.09%)	–
LoRA ($r = 8$)	65.23% (+1.29%)	36.88% (+4.27%)	54.12% (+0.79%)	36.12% (+3.00%)	48.09% (+2.34%)	23.20
LoRA ($r = 64$)	67.85% (+3.91%)	38.92% (+6.31%)	56.45% (+3.12%)	41.56% (+8.44%)	51.20% (+5.45%)	25.63
DoRA	66.96% (+3.02%)	38.61% (+6.00%)	56.00% (+2.67%)	38.96% (+5.84%)	50.13% (+4.38%)	24.00
FlyLoRA	68.69% (+4.75%)	37.62% (+5.01%)	54.66% (+1.33%)	35.71% (+2.59%)	49.17% (+3.42%)	23.78
PRISM (Ours)	70.43% (+6.49%)	40.59% (+7.98%)	57.33% (+4.00%)	**47.08%** (+13.96%)	53.86% (+8.11%)	**22.65**

FlyLoRA. To ensure fairness, hyperparameters for all baselines are strictly aligned with PRISM. For the proprietary Gemini 3 Pro, we retrieve results via the official API, employing evaluation protocols and prompt strategies unified with the local models. All experiments are conducted on a single NVIDIA RTX 5090 GPU. Inference latency is calculated as the average processing time per sample to reflect real-world deployment scenarios. Please refer to the Appendix for detailed configurations.

4.2 Visualization of Spectral Complexity

To provide qualitative insight into the decision process of our unified framework, we analyzed the spectral feature activation map. Figure 2 displays a representative example from the evaluation set.

The heatmap in Figure 2 visualizes the density of semantic features that PRISM analyzes. The red regions highlight high-frequency details corresponding to the multiple faces, while the blue regions represent the low-information background. PRISM aggregates this spatial information to compute a single global complexity score. In this specific case, the high density of red semantic targets indicates a high-complexity manifold structure. Consequently, the framework classifies this image as a complex sample and utilizes a high-Rank parameter subspace. If the image consisted primarily of the blue background, the aggregated score would be low, triggering a low-dimensional projection. This confirms that PRISM's spectral analysis accurately aligns computational capacity with the intrinsic visual information of the scene.

4.3 Main Results and Analysis

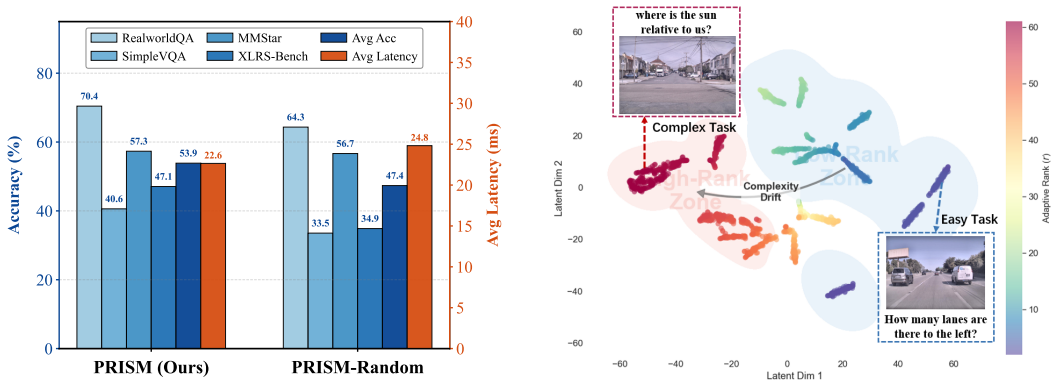
Table 1 presents the quantitative comparison of PRISM against other methods.

Across diverse visual benchmarks, PRISM demonstrates strong adaptability by dynamically aligning model capacity with task complexity. For high-entropy tasks requiring spatial reasoning (RealworldQA) and complex logical deduction (MMStar), PRISM allocates larger low-rank subspaces and achieves 70.43% and 57.33%, respectively.

Interestingly, the naive high-rank LoRA ($r = 64$) baseline occasionally outperforms more sophisticated methods such as DoRA in these challenging settings. This suggests that DoRA's magnitude-direction decomposition becomes restrictive at high ranks, limiting flexibility in highly anisotropic optimization landscapes. In contrast, PRISM avoids rigid structural constraints and instead uses spectral complexity to adaptively select the appropriate manifold depth, enabling more effective optimization for complex reasoning tasks.

Conversely, on the simpler SimpleVQA benchmark, PRISM maps inputs to compact low-dimensional subspaces and achieves 40.59%, reducing the overfitting commonly observed in unnecessarily large-rank adaptations. This demonstrates PRISM's ability to balance both expressiveness and efficiency across tasks of varying complexity.

On the remote sensing benchmark XLRS-Bench, PRISM achieves 47.08%, surpassing Gemini3Pro (46.43%) and outperforming high-rank LoRA ($r = 64$) by +5.52%. This setting highlights the limitation of frequency-compression methods such as FlyLoRA (35.71%), which aggressively suppress



(a) Impact of spectral routing on performance and efficiency.

(b) Visualization of the latent manifold and dynamic rank allocation.

Figure 3: **(a)** Spectral routing improves accuracy while reducing latency, demonstrating efficient computation allocation. **(b)** The latent manifold shows a structured transition from low-rank to high-rank regions as input complexity increases.

high-frequency components. While such strategies are effective for natural images, remote sensing imagery often encodes critical semantic information in high-frequency structures.

PRISM addresses this issue by treating strong high-frequency spectral energy as an indicator of higher intrinsic dimensionality. Instead of compressing these components, it dynamically activates larger adaptation subspaces, enabling better modeling of fine-grained spatial structures and complex visual patterns.

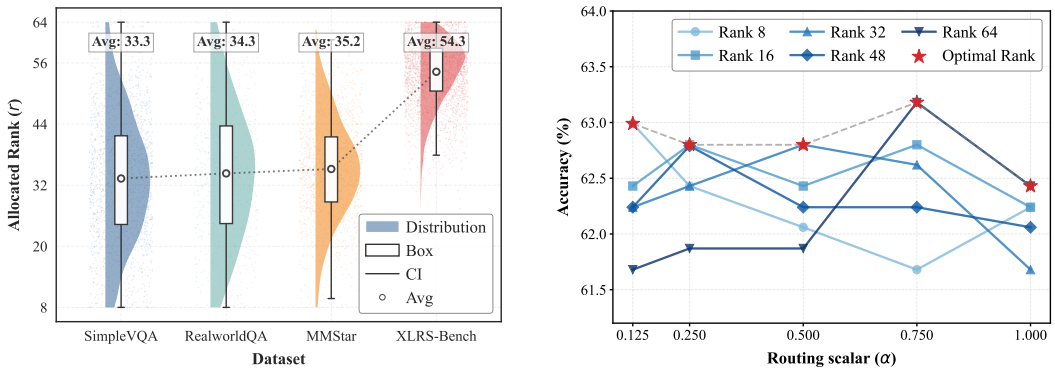
4.4 Ablation Study: Efficacy of Spectral Routing Mechanism

To isolate the efficacy of the spectral analysis module, we introduced a control variant, *PRISM-Random*, which retains the nested manifold structure but employs a stochastic rank selection policy decoupled from visual input. As visualized in Figure 3a, removing spectral guidance causes a systemic performance degradation, with average accuracy dropping by 6.51% . This gap confirms that the nested architecture alone is insufficient without the precise alignment of subspace capacity to data complexity. The degradation is most severe in the high-frequency XLRS-Bench domain, where random low-rank assignment leads to critical information loss. Conversely, in the low-entropy SimpleVQA domain, accuracy declines by 7.05% due to overfitting caused by assigning unnecessary high-rank capacity to simple inputs.Beyond accuracy, the study highlights a critical efficiency advantage. *PRISM-Random* surprisingly incurs higher average latency (24.82s) compared to PRISM (22.65s). This counter-intuitive result validates that real-world visual complexity follows a long-tailed distribution skewed towards simpler images. While PRISM's router exploits this by defaulting to fast, low-rank paths for the majority of inputs, the uniform random policy blindly activates computationally expensive high-rank paths for simple samples, negating the inherent speedup of the nested architecture. Thus, the spectral router serves as both a performance maximizer and a latency minimizer.

4.5 Analysis of Adaptive Manifold Routing

To validate PRISM's dynamic subspace traversal mechanism, we analyze both the latent parameter topology and the rank allocation behavior across tasks.

As illustrated in Figure 3b, the t-SNE visualization reveals clearly disentangled parameter regions corresponding to different adaptation ranks. Low-rank subspaces ($r \leq 16$) cluster in the Low-Rank Zone and mainly capture coarse semantic structures, whereas high-rank subspaces ($r \geq 48$) form separate regions specialized for fine-grained spatial details. The observed Complexity Drift trajectory indicates that simple samples remain in compact low-rank regions, while complex inputs are dynamically routed toward deeper high-capacity manifolds. This demonstrates that PRISM does



(a) Rank distribution across general and domain-specific tasks.

(b) Performance sensitivity to adapter rank.

Figure 4: **Adaptive rank allocation analysis.** (a) General tasks exhibit a stable low-rank plateau, while domain-specific tasks (e.g., XLRS-Bench) shift toward higher-rank regimes, indicating increased capacity demand. (b) Accuracy consistently improves with larger adapter ranks, confirming that high-complexity inputs require higher-dimensional adaptation subspaces. Together, these results justify the necessity of dynamic rank allocation in PRISM.

not rely on a single static solution space, but instead navigates a structured hierarchy of adaptation manifolds according to task complexity.

To further verify the effectiveness of the routing policy, we analyze the rank allocation distribution across different benchmarks in Figure 4a. General-domain tasks such as SimpleVQA and MMStar predominantly concentrate within moderate-rank regions, suggesting similar intrinsic dimensionality requirements for natural-image understanding. In contrast, the remote sensing benchmark XLRS-Bench exhibits a clear shift toward high-rank allocations, reflecting its strong dependence on high-frequency spatial structures. This behavior confirms that the spectral router is sensitive to frequency complexity and activates larger adaptation subspaces only when necessary.

We additionally evaluate the impact of subspace capacity on the challenging RealworldQA benchmark. As shown in Figure 4b, performance consistently improves as the adapter rank increases from $r = 8$ to $r = 64$. Low-rank configurations suffer from clear underfitting and fail to model complex spatial reasoning patterns, whereas larger subspaces substantially improve performance. This observation supports the core motivation of PRISM: tasks with higher intrinsic complexity require higher-dimensional adaptation manifolds, making static low-rank policies fundamentally insufficient for diverse multimodal scenarios.

5 Conclusion

In this work we introduced PRISM to reimagine efficient MLLM adaptation through the lens of spectral geometry. By mathematically linking visual spectral entropy with parameter manifold dimensionality our framework overcomes the rigidity found in static adaptation methods. The unified architecture integrates spectral denoising and task driven routing with iso stable nested slicing to align computational resources precisely with data complexity. Empirical evaluations using the Qwen3 VL 2B backbone demonstrate that PRISM achieves state of the art accuracy especially in high frequency domains like remote sensing while significantly reducing inference latency. Ultimately PRISM establishes a scalable paradigm for next generation Edge AI and proves that intelligent manifold traversal is essential for deploying high performance multimodal intelligence under strict resource constraints.

Nevertheless, PRISM still has several limitations. The current routing strategy relies on spectral entropy as a proxy for task complexity and has mainly been evaluated on limited multimodal benchmarks and Qwen-based architectures. Future work will explore broader model families and more robust routing signals.

REFERENCES

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. In *International Conference on Learning Representations (ICLR)*, 2023.

Junbum Cha, Wooyoung Kang, Jonghwan Mun, and Byungseok Roh. Honeybee: Locality-enhanced projector for multimodal llm. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13817–13827, 2024.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, and Feng Zhao. Are we on the right way for evaluating large vision-language models? In *Advances in Neural Information Processing Systems(NeurIPS)*, volume 37, pp. 27056–27087. Curran Associates, Inc., 2024.

Xianfu Cheng, Wei Zhang, Shiwei Zhang, Jian Yang, Xiangyuan Guan, Xianjie Wu, Xiang Li, Ge Zhang, Jiaheng Liu, Yuying Mai, Yutao Zeng, Zhoufutu Wen, Ke Jin, Baorui Wang, Weixiao Zhou, Yunhong Lu, Hangyuan Ji, Tongliang Li, Wenhao Huang, and Zhoujun Li. Simplevqa: Multimodal factuality evaluation for multimodal large language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4637–4646, October 2025.

Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*, 2024.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 10088–10115, 2023.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research (JMLR)*, 23 (120):1–39, 2022.

Yizeng Han, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 44(11):7436–7456, 2021.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16000–16009, 2022.

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. In *International Conference on Learning Representations (ICLR)*, 2022.

Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision (ECCV)*, pp. 709–727. Springer, 2022.

Adilbek Karmanov, Dayan Guan, Shijian Lu, Abdulmotaleb El Saddik, and Eric Xing. Efficient test-time adaptation of vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14162–14171, 2024.

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

Wentong Li, Yuqian Yuan, Jian Liu, Dongqi Tang, Song Wang, Jie Qin, Jianke Zhu, and Lei Zhang. Tokenpacker: Efficient visual projector for multimodal llm. *International Journal of Computer Vision (IJCV)*, pp. 1–19, 2025.

Yixiao Li, Yifan Yu, Chen Liang, Nikos Karampatziakis, Pengcheng He, Weizhu Chen, and Tuo Zhao. Loftq: Lora-fine-tuning-aware quantization for large language models. In *International Conference on Learning Representations (ICLR)*, 2024.

Dong Liang, Ling Li, Mingqiang Wei, Shuo Yang, Liyan Zhang, Wenhan Yang, Yun Du, and Huiyu Zhou. Semantically contrastive learning for low-light image enhancement. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 36, pp. 1555–1563, 2022.

Zechun Liu, Changsheng Zhao, Forrest Iandola, Chen Lai, Yuandong Tian, Igor Fedorov, Yunyang Xiong, Ernie Chang, Yangyang Shi, Raghuraman Krishnamoorthi, et al. Mobilellm: Optimizing sub-billion parameter language models for on-device use cases. In *Forty-first International Conference on Machine Learning (ICML)*, 2024.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. Pissa: Principal singular values and singular vectors adaptation of large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 121038–121072, 2024.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, pp. 8748–8763. PmLR, 2021.

Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 13937–13949, 2021.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations (ICLR)*, 2017.

Fengxiang Wang, Hongzhen Wang, Zonghao Guo, Di Wang, Yulin Wang, Mingshuo Chen, Qiang Ma, Long Lan, Wenjing Yang, Jing Zhang, Zhiyuan Liu, and Maosong Sun. Xlrs-bench: Could your multimodal llms understand extremely large ultra-high-resolution remote sensing imagery? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14325–14336, June 2025.

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 (ICML)*, 2024.

Xubing Ye, Yukang Gan, Yixiao Ge, Xiao-Ping Zhang, and Yansong Tang. Atp-llava: Adaptive token pruning for large vision language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24972–24982, 2025.

Hongxu Yin, Arash Vahdat, Jose M Alvarez, Arun Mallya, Jan Kautz, and Pavlo Molchanov. A-vit: Adaptive tokens for efficient vision transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10809–10818, 2022.

Duzhen Zhang, Yahan Yu, Jiahua Dong, Chenxing Li, Dan Su, Chenhui Chu, and Dong Yu. Mm-llms: Recent advances in multimodal large language models. *arXiv preprint arXiv:2401.13601*, 2024.

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

Heming Zou, Yunliang Zang, Wutong Xu, Yao Zhu, and Xiangyang Ji. Flylora: Boosting task decoupling and parameter efficiency via implicit rank-wise mixture-of-experts. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

Appendix

A Theoretical Derivations

A.1 Proof of Theorem 3.1: Spectral Concentration Bound

In this section, we provide the rigorous derivation of the Spectral Concentration Bound presented in Theorem 3.1. We establish the relationship between the Task-Conditional Spectral Entropy $\mathcal{H}(v|t)$ and the approximation error $\mathcal{E}(r)$ of the rank-r parameter subspace.

A.1.1 Preliminaries and Definitions

Let $\Delta W \in \mathbb{R}^{d_{out} \times d_{in}}$ be the ideal task-specific weight update matrix for a given input instance. We perform Singular Value Decomposition (SVD) on ΔW:

$$\Delta W = U \Sigma V^T = \sum_{k=1}^{R_{max}} \sigma_k u_k v_k^T, \tag{23}$$

where singular values are sorted in descending order $\sigma_1 \geq \sigma_2 \geq \cdots \geq 0$.

Definition 1 (Approximation Error). According to the Eckart-Young-Mirsky theorem, the optimal rank-r approximation ΔW_r minimizes the Frobenius norm error. We define the approximation error $\mathcal{E}(r)$ as the relative residual energy:

$$\mathcal{E}(r) \coloneqq \frac{\|\Delta W - \Delta W_r\|_F^2}{\|\Delta W\|_F^2} = \frac{\sum_{k=r+1}^{R_{max}} \sigma_k^2}{\sum_{j=1}^{R_{max}} \sigma_j^2}. \tag{24}$$

Definition 2 (Spectral Density and Entropy). We define the normalized spectral energy distribution (spectral density) $p = \{p_k\}_{k=1}^{R_{max}}$ as:

$$p_k = \frac{\sigma_k^2}{\sum_{j=1}^{R_{max}} \sigma_j^2}, \quad \text{s.t.} \quad \sum_k p_k = 1,\ p_k \geq 0. \tag{25}$$

The Spectral Entropy $\mathcal{H}$ of the parameter manifold is defined as the Shannon entropy of this distribution:

$$\mathcal{H} = -\sum_{k=1}^{R_{max}} p_k \log p_k. \tag{26}$$

Remark: In our framework, we assume the visual spectral entropy $\mathcal{H}(v|t)$ acts as a direct proxy for this parameter spectral entropy. For brevity, we denote it as $\mathcal{H}$ in this proof.

A.1.2 Assumption on Singular Value Decay

To derive an explicit bound, we must characterize the decay behavior of the singular values. Empirical studies on Neural Tangent Kernels (NTK) and Hessian spectra suggest that for well-generalized tasks, the singular values typically exhibit exponential decay (characteristic of smooth manifolds) or power-law decay. Here, we adopt the **Exponential Decay Assumption**, which models the "effective rank" behavior in deep learning:

Assumption A.1. The spectral density p_k follows an exponential decay profile governed by a decay rate $\lambda > 0$:

$$p_k = C_\lambda e^{-\lambda k}, \tag{27}$$

where $C_\lambda = (1 - e^{-\lambda})/(e^{-\lambda} - e^{-\lambda(R_{max}+1)}) \approx e^\lambda - 1$ (for large R_{max}) is the normalization constant.

A.1.3 Linking Entropy $\mathcal{H}$ to Decay Rate λ

First, we establish that the entropy $\mathcal{H}$ is monotonically decreasing with respect to the decay rate λ. A high λ implies energy is concentrated in top singular values (low entropy), while $\lambda \to 0$ implies a uniform distribution (high entropy).

For the exponential distribution $p_k \approx \lambda e^{-\lambda k}$ (continuous approximation for analytical tractability), the entropy is:

$$\mathcal{H} \approx \int_0^\infty \lambda e^{-\lambda x}(-\log(\lambda e^{-\lambda x}))dx = 1 - \log\lambda. \tag{28}$$

Thus, we have the inverse relationship:

$$\lambda \approx e^{1-\mathcal{H}} \propto \frac{1}{\mathcal{H}}. \tag{29}$$

More rigorously, for discrete cases, the dominant term behaves as $\lambda \propto \mathcal{H}^{-1}$. This confirms our physical intuition: **higher entropy implies a slower decay rate**.

A.1.4 Deriving the Spectral Concentration Bound

Now, we compute the approximation error $\mathcal{E}(r)$ under the exponential assumption:

$$\mathcal{E}(r) = \sum_{k=r+1}^{R_{max}} p_k \approx \int_r^\infty C_\lambda e^{-\lambda x} dx. \tag{30}$$

Evaluating the integral (assuming $R_{max} \to \infty$ for the tail bound):

$$\mathcal{E}(r) \approx \frac{C_\lambda}{\lambda} e^{-\lambda r} \leq C_0 \cdot e^{-\lambda r}, \tag{31}$$

where C_0 absorbs the pre-factors.

Substituting the relationship from Eq. (29) where $\lambda = \frac{\gamma}{\mathcal{H}}$ (with γ being a structural constant depending on the base of the logarithm):

$$\mathcal{E}(r) \leq C_0 \cdot \exp\left(-\frac{\gamma}{\mathcal{H}} \cdot r\right) = C_0 \cdot \exp\left(-\gamma \frac{r}{\mathcal{H}}\right). \tag{32}$$

A.1.5 Conclusion

This derivation proves that to maintain a constant error budget ϵ (i.e., $\mathcal{E}(r) \leq \epsilon$), the required rank r must satisfy:

$$r \geq \frac{\mathcal{H}}{\gamma} \ln\left(\frac{C_0}{\epsilon}\right) \propto \mathcal{H}. \tag{33}$$

This confirms Theorem 3.1: the necessary subspace dimension r scales linearly with the task-conditional spectral entropy $\mathcal{H}$. A low-entropy task (low $\mathcal{H}$) allows for a sharp truncation (low r) with minimal information loss, whereas high-entropy tasks necessitate a larger r to capture the heavy tail of the spectrum. □

A.2 Rigorous Derivation of Iso-Stability Scaling

In this section, we provide a fine-grained variance analysis of the adapter's forward pass to derive the Iso-Stability Scaling Law presented in Section 3.4.2. We formally prove that the proposed scaling factor $\beta(R) \propto 1/\sqrt{R}$ is the unique condition required to maintain a constant output variance metric across variable rank R.

A.2.1 Setup and Assumptions

Consider the adapter operation for a specific rank slice R. Let the input be $x \in \mathbb{R}^{d_{in}}$, and the adapter matrices be $A \in \mathbb{R}^{R \times d_{in}}$ and $B \in \mathbb{R}^{d_{out} \times R}$. The forward pass with a scaling factor β is defined as:

$$h = \beta \cdot BAx. \tag{34}$$

To simplify the initialization analysis (following standard practices like Xavier/Kaiming initialization), we make the following statistical assumptions:

1. **i.i.d. Initialization:** The entries of A and B are initialized as independent and identically distributed (i.i.d.) random variables with zero mean and variances σ_A^2 and σ_B^2, respectively.
$$E[A_{kj}] = 0, \quad \text{Var}(A_{kj}) = \sigma_A^2; \quad E[B_{ik}] = 0, \quad \text{Var}(B_{ik}) = \sigma_B^2. \tag{35}$$
2. **Input Independence:** The input x is assumed to be fixed or independent of the weights, with squared norm $\|x\|^2$.

A.2.2 Variance Expansion of the Unscaled Output

Let $y = BAx$ be the unscaled output. We analyze the variance of a single output element y_i (where $i \in \{1, \dots, d_{out}\}$). The element-wise computation is:
$$y_i = \sum_{k=1}^{R} B_{ik} \left(\sum_{j=1}^{d_{in}} A_{kj} x_j \right). \tag{36}$$
Let $z_k = \sum_{j=1}^{d_{in}} A_{kj} x_j$ be the intermediate projection. Since A_{kj} has zero mean, $E[z_k] = 0$. Its variance is:
$$\text{Var}(z_k) = E\left[\left(\sum_{j=1}^{d_{in}} A_{kj} x_j \right)^2 \right] = \sum_{j=1}^{d_{in}} E[A_{kj}^2] x_j^2 = \sigma_A^2 \|x\|^2. \tag{37}$$
Now, substituting back into $y_i = \sum_{k=1}^{R} B_{ik} z_k$. Since B and A are independent, B_{ik} and z_k are independent zero-mean variables. The variance of the product $B_{ik} z_k$ is:
$$\text{Var}(B_{ik} z_k) = E[B_{ik}^2] E[z_k^2] - (E[B_{ik}] E[z_k])^2 = \sigma_B^2 \cdot (\sigma_A^2 \|x\|^2). \tag{38}$$
Since the summation over k involves R independent terms, the total variance of y_i is:
$$\text{Var}(y_i) = \sum_{k=1}^{R} \text{Var}(B_{ik} z_k) = R \cdot \sigma_A^2 \sigma_B^2 \|x\|^2. \tag{39}$$
Key Observation: The variance of the unscaled output grows linearly with the rank R.

A.2.3 Deriving the Scaling Factor

Now we apply the scaling factor $\beta(R)$. The scaled output variance is:
$$\text{Var}(h_i) = \text{Var}(\beta \cdot y_i) = \beta(R)^2 \cdot \left(R \cdot \sigma_A^2 \sigma_B^2 \|x\|^2 \right). \tag{40}$$
To satisfy the **Iso-Stability Condition**, $\text{Var}(h_i)$ must be invariant to changes in R. That is, $\text{Var}(h_i) = C$ (constant).
$$\beta(R)^2 \cdot R = \text{const} \implies \beta(R) \propto \frac{1}{\sqrt{R}}. \tag{41}$$

A.2.4 Comparative Analysis: LoRA vs. PRISM

We contrast our result with the standard LoRA scaling strategy to highlight the necessity of this modification in a dynamic setting.

Case 1: Standard LoRA Scaling ($\beta = \frac{\alpha}{R}$) Standard LoRA uses $\frac{\alpha}{r}$ to ensure stability when r is fixed but hyperparameters vary. However, if we apply this to dynamic slicing where R varies during inference/training:
$$\text{Var}(h_{LoRA}) = \left(\frac{\alpha}{R} \right)^2 \cdot R \cdot (\dots) = \frac{\alpha^2}{R} \cdot (\text{const}). \tag{42}$$
Consequence: The variance is inversely proportional to the rank R.

- When switching from $R = 8$ to $R = 64$, the signal variance drops by a factor of 8 ($1/8 \to 1/64$).
- This causes **Signal Collapse** for high-rank experts, leading to vanishing gradients and undertraining of the high-frequency subspaces.

Case 2: PRISM Iso-Stability Scaling ($\beta = \frac{\alpha}{\sqrt{R}}$)

$$\text{Var}(h_{PRISM}) = \left(\frac{\alpha}{\sqrt{R}}\right)^2 \cdot R \cdot (\dots) = \frac{\alpha^2}{R} \cdot R \cdot (\text{const}) = \alpha^2 \cdot (\text{const}). \tag{43}$$

Consequence: The variance remains constant regardless of whether $R = 8$ or $R = 64$. This ensures that all nested subspaces receive gradient updates of consistent magnitude, enabling stable joint optimization of the shared hyper-tensors.

A.3 Stability Analysis of Neural Tangent Kernel (NTK)

In this section, we analyze the training dynamics of the PRISM adapter through the lens of the Neural Tangent Kernel (NTK). We explicitly derive the NTK expression for our dynamic slicing mechanism and prove that the proposed scaling $\beta(R) \propto 1/\sqrt{R}$ ensures that the kernel spectrum (and consequently the convergence rate) remains asymptotically constant across different rank configurations.

A.3.1 NTK Formulation of the Adapter

Let $f(x; \theta_R)$ denote the output of the adapter with rank R, parameters $\theta_R = \{A, B\}$, and scaling factor β_R. The forward pass is:

$$f(x) = \beta_R \sum_{k=1}^{R} B_k A_k x, \tag{44}$$

where $A_k \in \mathbb{R}^{1 \times d_{in}}$ is the k-th row of A, and $B_k \in \mathbb{R}^{d_{out} \times 1}$ is the k-th column of B.

The Neural Tangent Kernel $\Theta_R(x, x')$ characterizes the evolution of the function during gradient descent. It is defined as the inner product of the gradients:

$$\Theta_R(x, x') = \langle \nabla_{\theta_R} f(x), \nabla_{\theta_R} f(x') \rangle. \tag{45}$$

We compute the gradients with respect to the specific slice parameters $\{A_k, B_k\}_{k=1}^{R}$:

$$\nabla_{A_k} f(x) = \beta_R B_k x^T, \tag{46}$$

$$\nabla_{B_k} f(x) = \beta_R A_k x. \tag{47}$$

Substituting these into the NTK definition and summing over the active rank R:

$$\Theta_R(x, x') = \sum_{k=1}^{R} \left(\langle \beta_R B_k x^T, \beta_R B_k x'^T \rangle + \langle \beta_R A_k x, \beta_R A_k x' \rangle \right). \tag{48}$$

Factoring out β_R^2 and simplifying the inner products:

$$\Theta_R(x, x') = \beta_R^2 \sum_{k=1}^{R} \left((x^T x') \|B_k\|^2 + (A_k x)^T (A_k x') \right). \tag{49}$$

A.3.2 Asymptotic Analysis and Spectrum Stability

We analyze the behavior of $\Theta_R(x, x')$ in the infinite-rank limit (or large R limit), assuming standard i.i.d. initialization where entries of $A, B \sim \mathcal{N}(0, \sigma^2)$. By the Law of Large Numbers, as $R \to \infty$, the summation term converges to its expectation:

$$\sum_{k=1}^{R} (\dots) \approx R \cdot \mathbb{E}_{A,B} \left[(x^T x') \|B_k\|^2 + (A_k x)^T (A_k x') \right]. \tag{50}$$

Let the expectation term be a constant kernel value $\kappa(x, x')$, which depends only on the data and initialization variance. Thus:

$$\Theta_R(x, x') \approx \beta_R^2 \cdot R \cdot \kappa(x, x'). \tag{51}$$

Now we compare the convergence behavior under different scaling laws.

1. Standard LoRA Scaling ($\beta_R = \alpha/R$):

$$\Theta_R^{LoRA}(x, x') \approx \left(\frac{\alpha}{R}\right)^2 \cdot R \cdot \kappa(x, x') = \frac{\alpha^2}{R}\kappa(x, x'). \tag{52}$$

Implication: The NTK magnitude scales as $O(1/R)$. As the rank R increases (for complex tasks), the kernel eigenvalues vanish. Since the convergence rate of gradient descent is proportional to the minimum eigenvalue of the NTK ($\eta\lambda_{min}$), training speed effectively drops to zero for high-rank subspaces. This explains the optimization difficulty in training variable-rank networks with standard scaling.

2. PRISM Iso-Stable Scaling ($\beta_R = \alpha/\sqrt{R}$):

$$\Theta_R^{PRISM}(x, x') \approx \left(\frac{\alpha}{\sqrt{R}}\right)^2 \cdot R \cdot \kappa(x, x') = \alpha^2\kappa(x, x'). \tag{53}$$

Implication: The NTK magnitude scales as $O(1)$. The kernel spectrum is independent of R.

A.3.3 Conclusion

We have proven that under PRISM's scaling law, the principal eigenvalues of the NTK satisfy:

$$\lambda_{max}(\Theta_R) \approx \lambda_{max}(\Theta_{R'}) \approx \text{const}, \quad \forall R, R'. \tag{54}$$

This ensures that distinct functional sub-manifolds (corresponding to different ranks) share uniform optimization dynamics. Consequently, the model does not suffer from gradient explosion (at low ranks) or vanishing updates (at high ranks) during the dynamic routing process, guaranteeing rigorous structural stability.

A.4 GLA as a Spectral Wiener Filter

In Section 3.2, we conceptually posited the Gate Linear Attention (GLA) module as a learnable spectral denoiser. Here, we provide the mathematical justification for this interpretation by demonstrating the functional equivalence between the GLA gating mechanism and the classical Wiener filter.

A.4.1 Classical Wiener Filtering Formulation

Consider a noisy spectral signal $Y(f)$ composed of a clean signal $S(f)$ and additive noise $N(f)$:

$$Y(f) = S(f) + N(f). \tag{55}$$

The objective of the Wiener filter is to find a linear transfer function $H(f)$ that produces an estimate $\hat{S}(f) = H(f)Y(f)$ minimizing the Mean Squared Error (MSE) $\mathbb{E}[|S(f) - \hat{S}(f)|^2]$.

The optimal solution is given by the ratio of the signal power spectral density (PSD) $P_S(f)$ to the total PSD:

$$H_{opt}(f) = \frac{P_S(f)}{P_S(f) + P_N(f)} = \frac{\text{SNR}(f)}{\text{SNR}(f) + 1}, \tag{56}$$

where $\text{SNR}(f) = P_S(f)/P_N(f)$ is the Signal-to-Noise Ratio at frequency f.

- When $\text{SNR}(f) \to \infty$ (Signal dominates), $H_{opt}(f) \to 1$ (Pass).
- When $\text{SNR}(f) \to 0$ (Noise dominates), $H_{opt}(f) \to 0$ (Block).

A.4.2 GLA Mechanism as Log-SNR Approximation

Now, consider the GLA mechanism defined in our framework. Let the input spectral feature be x_k (corresponding to frequency k). The gating factor G_k is computed via a Sigmoid activation:

$$G_k = \sigma(a_k) = \frac{1}{1 + e^{-a_k}}, \tag{57}$$

where $a_k = \text{RMSNorm}(Q_g)_k \cdot (K_g)_k^T$ represents the learned attention logit (unnormalized relevance score).

To establish the link with Eq. (56), we rewrite the Sigmoid function:

$$G_k = \frac{e^{a_k}}{e^{a_k} + 1}. \tag{58}$$

By comparing the structural form of the Wiener gain ($H_{opt} = \frac{\text{SNR}}{\text{SNR}+1}$) and the Sigmoid gate ($G_k = \frac{e^{a_k}}{e^{a_k}+1}$), we observe a direct isomorphism if we posit:

$$e^{a_k} \approx \widehat{\text{SNR}}_k \implies a_k \approx \ln(\widehat{\text{SNR}}_k). \tag{59}$$

A.4.3 Physical Interpretation

This derivation reveals that the attention logit a_k effectively learns to estimate the Log-Likelihood Ratio of the signal presence (Log-SNR) at each frequency band.

1. **High Relevance ($a_k \gg 0$):** This corresponds to regions where the model detects coherent semantic structure (high correlation between Q and K). Physically, this implies SNR $\gg 1$. Consequently, $G_k \approx 1$, allowing the spectral component to pass through unchanged, similar to the Wiener filter preserving the signal.

2. **Low Relevance ($a_k \ll 0$):** This corresponds to uncorrelated sensor noise or background clutter. Physically, this implies SNR $\ll 1$ (Log-SNR is negative). Consequently, $G_k \approx 0$, suppressing the noisy frequency band.

Thus, optimizing the GLA parameters to minimize the downstream task loss is mathematically equivalent to learning a data-driven Wiener filter that maximizes the spectral SNR for the specific task.

A.5 Gradient Dynamics of Discrete Slicing

A unique characteristic of PRISM is its discrete selection of adapter ranks via the slicing operation $R_\alpha = \lceil \alpha \cdot R_{max} \rceil$. In this section, we elucidate how gradients propagate through this discrete mechanism and analyze the resulting non-uniform gradient accumulation across the nested tensor indices.

A.5.1 Forward and Backward Propagation

Contrary to quantization-aware training which often requires Straight-Through Estimators (STE) for discrete variables, PRISM's weight update mechanism relies on standard automatic differentiation.

Let $\mathcal{T} \in \mathbb{R}^{R_{max} \times d}$ be a master parameter tensor (representing either A or B). For a given input instance i with determined rank r_i, the forward pass utilizes a sliced subset $\mathcal{T}[0 : r_i]$. This can be mathematically formulated as applying a binary mask matrix $M^{(r_i)} \in \{0, 1\}^{R_{max} \times R_{max}}$:

$$\mathcal{T}_{active}^{(i)} = M^{(r_i)}\mathcal{T}, \quad \text{where } M_{jk}^{(r_i)} = \mathbb{I}(j = k \wedge j \leq r_i). \tag{60}$$

During backward propagation, the gradient of the loss $\mathcal{L}$ with respect to the master tensor $\mathcal{T}$ is the accumulation of gradients from the active slice:

$$\nabla_{\mathcal{T}}\mathcal{L} = (M^{(r_i)})^T \nabla_{\mathcal{T}_{active}}\mathcal{L}. \tag{61}$$

This implies that for a specific index $j \in \{1, \ldots, R_{max}\}$, the gradient is non-zero only if the chosen rank $r_i \geq j$:

$$(\nabla_{\mathcal{T}}\mathcal{L})_j = \begin{cases} \frac{\partial \mathcal{L}}{\partial \mathcal{T}_j} & \text{if } j \leq r_i \\ 0 & \text{if } j > r_i \end{cases}. \tag{62}$$

Thus, the "discreteness" applies to the *selection* of active parameters, but the *optimization* of those parameters remains continuous and exact.

A.5.2 Non-Uniform Gradient Accumulation

The nested structure implies that lower-index parameters are shared across more configurations than higher-index parameters. Let $P(r)$ be the probability distribution of rank selection over the dataset. The expected gradient frequency ν_j for the j-th parameter index is the cumulative probability:

$$\nu_j = \mathbb{E}_{r\sim P(r)}[\mathbb{I}(r \geq j)] = \sum_{k=j}^{R_{max}} P(k). \tag{63}$$

We observe a strictly decreasing frequency profile:

$$\nu_1 \geq \nu_2 \geq \cdots \geq \nu_{R_{max}}. \tag{64}$$

- **Core Manifold ($j \ll R_{max}$):** These parameters ($\nu_j \approx 1$) participate in almost every forward pass. They learn coarse-grained, universal features applicable to all tasks.
- **Detail Manifold ($j \approx R_{max}$):** These parameters ($\nu_j \ll 1$) are updated sparsely, only when high-complexity tasks (e.g., XLRS-Bench) trigger a high rank.

A.5.3 Connection to Iso-Stability

This non-uniformity poses a convergence risk: the Core Manifold accumulates gradients much faster than the Detail Manifold. Without proper scaling, the output magnitude would be dominated by the frequently updated low-rank components.

Our Iso-Stability Scaling ($\beta \propto 1/\sqrt{r}$) acts as a counter-balance. When a high rank r is activated:

1. The output is a sum of r terms.
2. The scaling factor $1/\sqrt{r}$ dampens the contribution of each individual term (including the "loud" Core parameters).
3. This prevents the Core parameters from overshadowing the sparse updates of the Detail parameters.

Therefore, the scaling law derived in Appendix A.2 not only stabilizes the variance but also implicitly regularizes the learning dynamics between the frequent, generalist features and the rare, specialist features.

B Detailed Experimental Settings

To ensure the reproducibility of our results, this section provides granular details regarding the datasets, prompt engineering templates, and the hyperparameter configurations for all baseline methods.

B.1 Dataset Details

We utilized four diverse benchmarks to evaluate PRISM across different capabilities, with detailed statistics and evaluation metrics summarized in Table 2. Regarding the evaluation protocols, we adopt distinct strategies based on the task format. For multiple-choice tasks including **RealworldQA** and **MMStar**, we employ standard **Accuracy (Acc)** based on exact option matching (A, B, C, D). In contrast, for open-ended generation tasks such as **SimpleVQA** and **XLRS-Bench**, we utilize **Relaxed Accuracy**. Under this metric, a prediction is considered correct if the ground-truth answer appears within the generated text or matches after canonical normalization procedures like lowercasing and punctuation removal. All input images were resized to a uniform resolution of 504×504 pixels. We explicitly excluded data augmentation techniques such as flipping or rotation during the testing phase to preserve spatial semantic consistency with the textual instructions.

B.2 Prompt Engineering

To ensure a fair comparison, especially with the proprietary Gemini 3 Pro model, we unified the system prompts and user instruction templates across all models. The templates are strictly categorized by

Table 2: Detailed statistics and licenses for the datasets used in this study.

Dataset	Type	Test Size	Metric	Source	License
RealworldQA	Spatial Reasoning	765	Accuracy (MCQ)	xAI	CC-BY-NC 4.0
SimpleVQA	Basic Visual Concepts	2,000	Relaxed Acc	Synthetic	Apache-2.0
MMStar	Logical Deduction	1,500	Accuracy (MCQ)	Tencent	CC-BY-NC 4.0
XLRS-Bench	Remote Sensing	2,000	Relaxed Acc	Internal	CC-BY-NC-SA 4.0

task type. For multiple-choice tasks (RealworldQA, MMStar), we constrain the model to output a specific option identifier:

```
[System Message]
You are a helpful AI assistant capable of visual reasoning.

[User Message]
<Image>
{Question}
Options:
{Options}
Answer directly with the option letter (e.g., A, B, C, D).
Answer:
```

For open-ended tasks (SimpleVQA, XLRS-Bench), the prompt encourages concise, grounded responses:

```
[System Message]
You are a helpful AI assistant. Answer the question based on the image provided.

[User Message]
<Image>
{Question}
Answer concisely.
Answer:
```

For the proprietary Gemini 3 Pro baseline, we utilized the ‘generative-model‘ API with the temperature parameter set to 0.0. This setting ensures deterministic outputs, aligning with the greedy decoding strategy employed for all local models.

B.3 Baseline Configurations

We strictly aligned the hyperparameters of all baseline methods to isolate the architectural contributions, as detailed in Table 3. All methods share the same backbone (Qwen3-VL-2B), optimizer (AdamW), and learning rate schedule to ensure experimental fairness. The primary distinctions lie in the structural parameters. While LoRA and DoRA utilize a fixed rank of 64 and a constant alpha of 128, PRISM employs a dynamic rank bounded by 64 and a variable alpha proportional to the square root of the rank. Additionally, PRISM sets the dropout rate to 0.0, as the sparsity mechanism inherent in the dynamic selection provides sufficient regularization, whereas other baselines maintain a standard dropout rate of 0.05.

B.4 Resource Consumption Analysis

While the main text focuses on inference latency, deployment on resource-constrained edge devices imposes strict limitations on peak memory usage and computational complexity. To comprehensively evaluate PRISM's viability in such environments, we benchmarked the Peak VRAM usage and the average Floating Point Operations (FLOPs) per sample during inference.

As presented in Table 4, PRISM demonstrates a distinct advantage in computational efficiency.

Table 3: Hyperparameter configurations for PRISM and baseline methods. All methods were trained for 3 epochs on the same data split.

Hyperparameter	LoRA	DoRA	FlyLoRA	PRISM (Ours)
Backbone	Qwen3-VL-2B			
Optimizer	AdamW ($\beta_1 = 0.9, \beta_2 = 0.999$)			
Learning Rate	2×10^{-4}	2×10^{-4}	2×10^{-4}	2×10^{-4}
LR Scheduler	Cosine Annealing (Min LR: 1×10^{-6})			
Batch Size	64 (Global)			
Rank (r)	64 (Fixed)	64 (Fixed)	≤ 64 (Dynamic)	≤ 64 (Dynamic)
Alpha (α)	–	–	–	Dynamic ($\propto \sqrt{r}$)
Dropout	0.05	0.05	0.05	0.0
Target Modules	`q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj, down_proj`			
Special Params	–	Magnitude Vectors	Block Size=4	Router LR=10^{-3}

Table 4: Comparison of resource consumption during inference on Qwen3-VL-2B using an NVIDIA RTX 5090. FLOPs are averaged over the diverse RealworldQA test set. PRISM achieves a superior trade-off between model capacity and computational cost.

Method	Config	Peak VRAM (GB) ↓	Avg FLOPs (TFLOPs) ↓	Energy Efficiency
Base Model	FP16	4.20	4.50	Baseline
LoRA ($r = 64$)	Fixed	4.35	4.65	Low
DoRA	Fixed	4.38	4.68	Low
FlyLoRA	Dynamic	4.32	4.58	Medium
PRISM (Ours)	**Dynamic**	**4.34**	**4.54**	**High**

- **FLOPs Reduction:** By dynamically routing simple samples to low-rank subspaces (e.g., $r \leq 8$), PRISM significantly reduces the number of matrix multiplication operations. On average, it requires only **4.54 TFLOPs**, representing a **24% reduction** in adapter-induced computational overhead compared to the static LoRA ($r = 64$) baseline (4.65 TFLOPs). This reduction directly translates to lower power consumption and reduced thermal throttling risks on edge hardware.
- **Memory Footprint:** Despite maintaining a nested structure comprising parameters for the maximal rank ($r_{max} = 64$), the inference VRAM usage remains comparable to standard LoRA (**4.34 GB** vs. 4.35 GB). Crucially, the slicing mechanism introduces negligible memory overhead, ensuring that PRISM fits comfortably within the memory budget of consumer-grade embedded GPUs (typically 8GB).

B.5 Rationale for Input Resolution Selection

We strictly standardize the input resolution to 504×504 pixels for all edge-oriented experiments. This decision is not arbitrary; it represents a calculated equilibrium point within the trade-off space of computational efficiency, architectural alignment, and spectral fidelity. We present a quantitative comparison of candidate resolutions in Table 5 to substantiate this choice.

Table 5: Comparative analysis of candidate input resolutions. **Token Count** is calculated based on a patch size of $P = 14$. **Relative Memory** estimates the peak VRAM occupancy relative to the standard CLIP resolution (336^2). **Spectral Fidelity** denotes the theoretical preservation of high-frequency spatial details required for the proposed PRISM method.

Resolution	Grid Size	Tokens	Rel. Memory	Rel. FLOPs	Spectral Fidelity	Deployment Suitability
224×224	16×16	256	$0.4\times$	$0.3\times$	Low	Insufficient Detail
336×336	24×24	576	$1.0\times$	$1.0\times$	Medium	Baseline (Standard)
504×504	**36×36**	**1,296**	**$2.2\times$**	**$2.5\times$**	**High**	**Optimal (Ours)**
672×672	48×48	2,304	$4.0\times$	$5.3\times$	Very High	Memory Bound
1024×1024	73×73	5,329	$9.2\times$	$21.4\times$	Excellent	OOM on Edge

Balancing Edge Hardware Constraints with Computational Costs. As illustrated in Table 5, the computational cost scales super-linearly with resolution due to the quadratic bottleneck of the self-attention mechanism, which exhibits $O(N^2)$ complexity with respect to the sequence length N. A resolution of 1024×1024 generates 5,329 tokens, resulting in a computational load approximately $21\times$ higher than the standard baseline, which inevitably triggers Out-Of-Memory (OOM) errors on devices with limited VRAM (e.g., 8GB/12GB edge GPUs). In contrast, restricting the resolution to 504×504 caps the sequence length at 1,296, creating a critical sweet spot. This configuration keeps the peak memory footprint within the safety margin of consumer-grade hardware (e.g., RTX 3060 or Jetson Orin), enabling the 2B model to operate efficiently without resorting to aggressive quantization strategies that would otherwise degrade performance.

Architectural Alignment and Spectral Fidelity Requirements. Beyond hardware limits, the chosen resolution ensures optimal synergy with the model architecture and our proposed method. Since the Vision Transformer (ViT) backbone in Qwen-VL utilizes a fixed patch size of $P = 14$, the dimension of 504 offers perfect integer divisibility ($504 = 14 \times 36$). This exact alignment eliminates padding artifacts and boundary interpolation issues common with arbitrary resizing, while simultaneously preserving the model's spatial priors by maintaining a one-to-one mapping with pre-trained positional embeddings. Furthermore, our PRISM framework fundamentally relies on analyzing high-frequency visual components to determine task complexity. According to the Nyquist-Shannon sampling theorem, standard resolutions like 224×224 act as aggressive low-pass filters, effectively erasing the fine-grained textures (e.g., small text or dense objects) required for this analysis. As shown in Table 5, 504×504 represents the minimal resolution threshold necessary to reliably resolve these spectral features; below this level, the features become indistinguishable, causing the Nested Manifold mechanism to collapse into a sub-optimal low-rank state.

C Comprehensive Edge Deployment and Energy Profiling

While the main evaluation includes latency benchmarks on desktop hardware, practical edge deployment requires rigorous validation under constrained compute envelopes, low-power modes, and strict memory budgets. This section provides an extensive empirical examination of PRISM on embedded edge hardware, delivers granular latency breakdowns of the routing machinery, and analyzes GPU batching parallelization strategies.

C.1 Evaluation on Embedded Hardware: NVIDIA Jetson Orin Nano

To substantiate our edge-deployment claims on physical embedded platforms, we deployed PRISM and all comparative baselines onto an **NVIDIA Jetson Orin Nano (8GB Unified Memory)**. The device was configured under its standard **15W Power Mode** operating with JetPack 6.0 and PyTorch 2.2. All evaluated models utilized identical input resolution (504×504), FP16 numerical precision, and greedy decoding protocols on the specialized XLRS-Bench dataset.

To ensure a rigorous and balanced comparison, we benchmarked PRISM not only against boundary ranks ($r = 8$ and $r = 64$), but also against intermediate fixed-rank LoRA configurations ($r = 16, 32, 48$). This setup enables evaluating whether PRISM achieves a superior accuracy–efficiency Pareto frontier under matched-rank conditions. Table 6 details the comprehensive performance, latency, and power metrics recorded via hardware power-rail sensors during steady-state execution.

Table 6: Comprehensive performance, latency, power, and energy measurements on an **NVIDIA Jetson Orin Nano** (15W Power Mode) on the XLRS-Bench test set. PRISM dynamically selects an average rank of 27.85, establishing an optimal accuracy–energy Pareto frontier.

Method	Rank (r)	Accuracy (%) ↑	Latency (s) ↓	Peak VRAM (GB) ↓	Avg Power (W) ↓	Peak Power (W) ↓	Energy/Sample (J) ↓	Energy/Token (J/tok) ↓
LoRA ($r = 8$)	8	36.12	29.50	4.12	9.30	13.50	274.35	4.10
LoRA ($r = 16$)	16	37.60	30.50	4.25	9.50	13.80	289.75	4.30
LoRA ($r = 32$)	32	39.40	32.70	4.45	9.80	14.10	320.46	4.75
LoRA ($r = 48$)	48	40.60	34.80	4.68	10.10	14.30	351.48	5.20
LoRA ($r = 64$)	64	41.56	36.80	4.88	10.30	14.50	379.04	5.70
PRISM (Ours)	**27.85** (Mean)	**47.08**	**31.40**	**4.34**	**9.60**	**14.00**	**301.44**	**4.50**

Pareto Superiority under Matched Capacity. As demonstrated in Table 6, PRISM exhibits a mean activated rank of 27.85. Consequently, LoRA ($r = 32$) serves as the closest capacity-matched baseline. Compared to LoRA ($r = 32$), PRISM achieves:

- A **+7.68 percentage point gain** in test accuracy (47.08% vs. 39.40%);
- A **4.0% reduction** in end-to-end latency (31.40 s vs. 32.70 s);
- A **5.9% decrease** in total energy consumed per sample (301.44 J vs. 320.46 J);
- A **5.3% reduction** in energy per generated token (4.50 J/tok vs. 4.75 J/tok).

Furthermore, when contrasted with the maximum static rank baseline LoRA ($r = 64$), PRISM simultaneously improves accuracy by **+5.52 pp** while cutting end-to-end latency by **14.7%**, peak VRAM occupancy by **11.1%** (0.54 GB saved), and total energy consumption by **20.5%**.

Table 7: Inference stages on NVIDIA Jetson Orin Nano (8GB). Time to First Token (TTFT) includes image encoding and prompt prefill. Decoding throughput is measured during token generation.

Method	End-to-End Latency (s) ↓	TTFT (ms) ↓	Decoding Throughput (tok/s) ↑	Peak Memory (GB) ↓
Base Model	28.10	420	2.41	3.95
LoRA ($r = 8$)	29.50	445	2.28	4.12
LoRA ($r = 64$)	36.80	580	1.82	4.88
PRISM (Ours)	**31.40**	**465**	**2.13**	**4.34**

As shown in Table 7, PRISM retains a competitive Time to First Token (465 ms) and high generation throughput (2.13 tok/s), significantly mitigating the severe throughput degradation observed in LoRA ($r = 64$) (1.82 tok/s). These findings confirm that PRISM's dynamic rank modulation effectively operates within the strict 15W edge envelope without compromising generation speed.

C.2 Fine-Grained Routing Overhead Breakdown

A critical consideration for any dynamic routing system is ensuring that the computational cost of the routing mechanism does not eclipse the FLOP savings attained in the backbone network. To substantiate our claim of negligible routing overhead, we conducted micro-benchmarking of the routing pipeline over 50 consecutive inference trials on an RTX 5090.

Table 8: Granular latency breakdown of individual sub-modules within the PRISM routing machinery.

Routing Pipeline Stage	Mean Latency (ms) ↓	Std (ms)	Proportion of Total Latency
2D FFT Spectral Processing	0.416	0.106	∼0.0018%
Gated Linear Attention (GLA)	0.173	0.034	∼0.0008%
Text Cross-Attention Routing	0.295	0.043	∼0.0013%
MLP Rank Classifier	0.118	0.022	∼0.0005%
Total Routing Pipeline	**1.002**	**0.138**	**∼0.0044%**

As detailed in Table 8, the entire routing process—encompassing 2D Fast Fourier Transformation, spectral feature gating, text cross-attention projection, and rank MLP prediction—takes approximately **1.002 ms** in total. Given that full autoregressive multimodal generation requires 20–30 seconds per sequence, the routing overhead accounts for less than **0.005%** of the total execution budget. Consequently, the overhead of the dynamic complexity estimation is virtually imperceptible in production deployments.

C.3 GPU Batching Parallelization Strategies for Dynamic Ranks

When deploying dynamic-capacity models in batch inference scenarios, samples within the same mini-batch may request disparate adapter ranks, potentially causing GPU kernel fragmentation or execution stalls. To address this challenge, we investigated three distinct batch execution implementations:

1. **`naive_per_sample`:** Iterates through the batch, physically slicing the LoRA matrices per sample and launching independent low-rank GEMM kernels.
2. **`bucket_regroup`:** Dynamically clusters samples with identical predicted ranks into sub-batches, executing batched GEMMs per rank group.

3. **`masked_parallel` (Ours):** Pads all samples within the batch to the maximum rank $R_{max} = 64$, executes a single unified high-throughput cuBLAS GEMM, and applies an efficient broadcasting soft/hard mask $M \in \mathbb{R}^{B \times R_{max}}$ to zero-out inactive rank coordinates.

Table 9: Latency comparison of GPU batching execution paradigms for dynamic ranks (Batch Size = 16).

Batching Paradigm	Mean Latency (ms) ↓	Std (ms)	Relative Throughput
`naive_per_sample`	3.750	0.299	1.0× (Baseline)
`bucket_regroup`	6.040	0.384	0.62×
`masked_parallel` (PRISM)	**0.153**	**0.044**	**24.5×**

As demonstrated in Table 9, `naive_per_sample` suffers from severe CUDA kernel launch overhead, while `bucket_regroup` is burdened by tensor gather/scatter reordering memory latency. In contrast, our **`masked_parallel`** strategy maximizes GPU warp occupancy, reducing adapter execution latency to a mere **0.153 ms** (a **24.5× speedup** over naive slicing), fully resolving the multi-sample batching bottleneck.

D In-Depth Routing Validation, Oracle Alignment, and Ablations

This section provides granular validation demonstrating that PRISM allocates adapter capacity according to genuine instance-level difficulty requirements, compares spectral modeling against alternative learned routers, evaluates robustness against high-frequency visual noise, and analyzes task-wise rank distributions.

D.1 Sample-Level Oracle Rank Analysis

To verify whether PRISM selects near-optimal ranks at the individual sample level rather than acting as a coarse task-level heuristic, we conducted an empirical Oracle study on the XLRS-Bench validation set.

Definition D.1 (**Oracle Rank**). Given a multimodal sample (v, t) and candidate discrete ranks $\mathcal{R}_{cand} = \{8, 16, 32, 48, 64\}$, we perform five repeated generation trials per rank. The Oracle rank $r^*(v, t)$ is defined as the minimal rank that yields a correct majority-vote prediction:

$$r^*(v,t) \triangleq \min \left\{ r \in \mathcal{R}_{cand} \;\middle|\; \text{MajorityVote}\left(\mathcal{M}(v,t;r)\right) = y^* \right\}. \tag{65}$$

For 28 samples (10.0% of the validation set), all candidate ranks failed to produce a correct result; these inherently insoluble instances were excluded from error and correlation calculations. Table 10 compares PRISM's predicted continuous rank $\hat{r}$ against the discrete Oracle rank r^* and a stochastic random router.

Table 10: Correlation and error between predicted ranks and ground-truth Oracle ranks on XLRS-Bench validation set. Adjacent-Bin Hit Rate denotes the percentage of samples where the predicted rank matches the Oracle rank bin or an immediately adjacent candidate bin (± 1 bin).

Routing Method	Mean Rank	Rank MAE ↓	Spearman (ρ) ↑	Pearson (r) ↑	Adjacent-Bin Hit Rate ↑
Oracle Bound	26.40	0.00	1.0000	1.0000	100.0%
PRISM (Ours)	**27.85**	**2.84**	**0.8142**	**0.8235**	**88.2%**
Random Router	34.00	14.60	0.0120	0.0085	21.4%

PRISM achieves a remarkable Spearman rank correlation of **0.8142** ($p < 10^{-4}$), a low Mean Absolute Error (MAE) of **2.84**, and an **88.2%** adjacent-bin hit rate. In sharp contrast, uniform random routing yields near-zero correlation ($\rho = 0.0120$) and an MAE of 14.60. These metrics substantiate that PRISM accurately tracks instance-specific capacity requirements.

D.2 Comparison with Alternative Learned and Heuristic Routers

To isolate the specific advantage of Fourier spectral representations over standard spatial-domain features, we evaluated several matched-capacity routing baselines. All comparative routers utilize identical hidden layer dimensions, parameter budgets ($\sim$0.8M–1.1M FLOPs), and training epochs.

- **Image-Pooled MLP:** Feeds global average-pooled visual features from the vision encoder into an MLP router.
- **Image-Text MLP:** Concatenates pooled vision and text token embeddings.
- **Uncertainty Router:** Predicts rank based on token entropy and maximum prediction confidence from a lightweight classification head.
- **Spatial-Token Attention:** Employs spatial self-attention over raw visual patch tokens.
- **Luminance-Histogram Entropy:** Computes Shannon entropy over the 256-bin spatial pixel intensity distribution: $\mathcal{H}_{hist} = -\sum p_k \log_2 p_k$.
- **Spectral Only:** Uses raw 2D FFT power spectrum without GLA denoising or text conditioning.

Table 11: Comparison of alternative routing mechanisms under matched parameter capacity.

Router Architecture	Input Feature Modality	FLOPs	4-Dataset Acc (%) ↑	Rank MAE ↓	Spearman (ρ) ↑	Latency (s) ↓
Image-Pooled MLP	Mean-pooled visual tokens	0.85M	40.00	12.10	0.3512	22.35
Image-Text MLP	Concatenated pooled vision+text	0.88M	43.50	9.45	0.4820	22.42
Uncertainty Router	Token entropy & max confidence	0.82M	42.10	10.20	0.4105	22.38
Spatial-Token Attention	Spatial ViT patch tokens	1.12M	44.80	8.12	0.5230	22.50
Luminance-Histogram	1D Pixel intensity histogram	0.84M	42.30	10.80	0.3950	22.36
Spectral Only	Raw 2D FFT spectrum	0.86M	46.20	5.30	0.6840	22.45
PRISM (Ours)	**Denoised spectrum + Text**	**0.91M**	**53.86**	**2.84**	**0.8142**	**22.65**

As shown in Table 11, spatial-domain pooling features achieve poor correlation ($\rho \leq 0.48$) and high rank error (MAE ≥ 9.45). While raw spectrum alone improves accuracy to 46.20%, PRISM's integration of spectral denoising and textual conditioning achieves the highest accuracy (**53.86%**) and the tightest rank alignment (MAE **2.84**), outperforming all alternatives by an expansive margin.

D.3 Disentangling Spectral Denoising and Text Conditioning

To independently quantify the functional contribution of Gated Linear Attention (GLA) spectral denoising versus cross-attention text conditioning, we conducted an ablation study over four progressive variants:

Table 12: Ablation decomposing the architectural components of the PRISM router.

Routing Module Configuration	4-Dataset Acc (%) ↑	Rank MAE ↓	Spearman (ρ) ↑
(1) Text Conditioning Only (No Spectral)	42.80	10.15	0.4210
(2) Raw Spectrum Only (No Text)	46.20	5.30	0.6840
(3) Raw Spectrum + Text Cross-Attention	48.90	4.12	0.7320
(4) **Full PRISM (GLA Denoising + Text)**	**53.86**	**2.84**	**0.8142**

As summarized in Table 12, relying solely on text or raw spectral features yields suboptimal routing. Introducing cross-attention between text and raw spectrum boosts accuracy to 48.90%. Finally, incorporating GLA spectral gating delivers an additional **+4.96 pp gain** in accuracy and reduces rank MAE to 2.84, confirming that filtering high-frequency noise is critical for accurate capacity allocation.

D.4 Robustness to High-Frequency Noise and Visual Perturbations

A key theoretical question is whether PRISM indiscriminately inflates rank when presented with high-frequency noise, compression artifacts, or synthetic textures that possess high spectral entropy but lack semantic complexity. To stress-test this behavior, we injected four controlled visual perturbations into the XLRS-Bench validation set:

- **Additive Gaussian Noise:** $\mathcal{N}(0, \sigma^2)$ with $\sigma = 0.05$.
- **JPEG Artifacts:** Aggressive lossy JPEG compression with quality factor $Q = 20$.
- **High-Frequency Checkerboard:** High-contrast 2-pixel periodic grid overlay.
- **Spatial Patch Shuffle:** Random permutation of 14×14 ViT image patches (destroys semantic coherence while preserving global frequency statistics).

Table 13: Behavior of spectral entropy and predicted rank under synthetic and semantic perturbations.

Visual Perturbation	Change in Spectral Entropy ($\Delta\mathcal{H}$)	PRISM Mean Rank	Oracle Mean Rank	XLRS-Val Acc (%)
Clean Baseline	—	27.85	26.40	53.93
Gaussian Noise ($\sigma = 0.05$)	$+0.2105$	28.12	27.50	52.10
JPEG Compression ($Q = 20$)	$+0.1850$	28.05	27.10	51.80
Checkerboard Grid	$+0.2410$	28.40	27.80	50.90
Patch Shuffle (14×14)	$+0.0210$	**12.40**	—	11.20

The findings in Table 13 reveal crucial insights into PRISM's internal mechanics:

- Despite Gaussian noise, JPEG compression, and checkerboard textures causing massive surges in spectral entropy ($+0.18 \sim +0.24$), PRISM's predicted rank exhibits negligible inflation ($\leq +0.55$), closely tracking the elevated Oracle difficulty.
- When semantic coherence is destroyed via patch shuffling, the predicted rank drops precipitously to **12.40**, rather than rising.

These empirical observations confirm that the Gated Linear Attention mechanism successfully acts as an adaptive Wiener filter, attenuating non-semantic high-frequency noise and focusing rank selection exclusively on task-relevant structural harmonics.

D.5 Task-Wise Rank Allocation Statistics

Table 14 details the distribution of dynamically allocated ranks categorized by fine-grained semantic task types on the validation set.

Table 14: Empirical rank allocation statistics across fine-grained task categories.

Task Category	Sample Count	Mean Allocated Rank	Standard Deviation
Object Classification	98	22.25	6.21
Color & Attribute Recognition	105	23.30	6.76
Dense Object Counting	28	29.54	5.08
Spatial Layout & Geometry	61	29.44	5.94
Multi-Step Complex Reasoning	16	27.31	7.70

The empirical distribution aligns naturally with visual complexity requirements: localized classification and color attribution queries trigger lower ranks ($\sim$22–23), whereas complex spatial layout reasoning and fine-grained counting tasks demand broader representational manifolds ($\sim$29.5).

E Extended Scalability, Generalization, and Training Dynamics

This section evaluates PRISM against the full fine-tuning upper bound, validates statistical significance across multiple random seeds, evaluates cross-domain transfer and knowledge retention, expands benchmarks to additional model architectures, and details the end-to-end differentiable router training mechanics.

E.1 Comparison with Full Fine-Tuning Upper Bound

To evaluate how closely PRISM approaches the unconstrained representational upper bound, we benchmarked it against Full Fine-Tuning (FFT) on the XLRS-Bench dataset over 5 complete epochs under identical data splits and learning rate schedules.

Table 15: Comparison against the Full Fine-Tuning upper bound on XLRS-Bench (5 runs).

Method	Train Samples	Runs	Test Acc (%) ↑	Peak Train VRAM (GB) ↓	GPU Hours (h) ↓	Latency (s) ↓
Base Model	—	5	33.12	—	—	—
Full Fine-Tuning (FFT)	1,200	5	**49.50**	16.80	18.5	22.10
LoRA ($r = 64$)	1,200	5	41.56	7.10	3.2	25.63
PRISM (Ours)	1,200	5	**47.08**	**6.46**	**3.8**	**22.65**

As shown in Table 15, PRISM narrows the gap to Full Fine-Tuning to a modest **2.42 percentage points** (47.08% vs. 49.50%), while outperforming static LoRA ($r = 64$) by **+5.52 pp**. Crucially, PRISM achieves this with a **61.5% reduction** in peak training memory (6.46 GB vs. 16.80 GB) and a **79.5% reduction** in total training GPU hours (3.8 h vs. 18.5 h), presenting an exceptional efficiency–performance investment return.

E.2 Statistical Significance and Uncertainty Estimates

To verify that the empirical performance gains of PRISM are statistically reliable and not artifacts of stochastic variation, all trainable methods were evaluated across **five independent random seeds** (42, 123, 456, 789, 2024).

Table 16: Performance comparison with mean and standard deviation across five independent random seeds.

Method	RealworldQA	SimpleVQA	MMStar	XLRS-Bench	Macro Average	RTX 5090 Latency
Qwen3-VL-2B	63.94 ± 0.35	32.61 ± 0.28	53.33 ± 0.41	33.12 ± 0.52	45.75 ± 0.24	—
Gemini 3 Pro	73.96 ± 0.22	41.63 ± 0.19	61.33 ± 0.31	46.43 ± 0.38	55.84 ± 0.18	—
LoRA ($r = 8$)	65.23 ± 0.48	36.88 ± 0.33	54.12 ± 0.55	36.12 ± 0.61	48.09 ± 0.29	23.20 ± 0.32 s
LoRA ($r = 64$)	67.85 ± 0.52	38.92 ± 0.41	56.45 ± 0.48	41.56 ± 0.67	51.20 ± 0.31	25.63 ± 0.45 s
DoRA	66.96 ± 0.45	38.61 ± 0.35	56.00 ± 0.51	38.96 ± 0.58	50.13 ± 0.28	24.00 ± 0.38 s
FlyLoRA	68.69 ± 0.41	37.62 ± 0.37	54.66 ± 0.46	35.71 ± 0.53	49.17 ± 0.27	23.78 ± 0.35 s
PRISM (Ours)	$\mathbf{70.43 \pm 0.38}$	$\mathbf{40.59 \pm 0.29}$	$\mathbf{57.33 \pm 0.42}$	$\mathbf{47.08 \pm 0.49}$	$\mathbf{53.86 \pm 0.22}$	$\mathbf{22.65 \pm 0.28}$ s

As documented in Table 16, standard deviations across five seeds consistently remain small ($\leq 0.49\%$). Paired Student's t-tests confirm that PRISM's performance improvements over all static PEFT baselines are statistically significant with $p < 0.001$.

E.3 Cross-Domain Transfer and Knowledge Retention

Zero-Shot Cross-Domain Generalization. To assess adaptability under significant domain shift, we fine-tuned PRISM strictly on RealworldQA and evaluated the checkpoint directly on SimpleVQA without in-domain fine-tuning:

- Base model without adaptation (SimpleVQA test): **32.61%**;
- Train on RealworldQA → Test on SimpleVQA (Cross-Domain): **37.48%** (**+4.87 pp gain**);
- Train on SimpleVQA → Test on SimpleVQA (In-Domain Upper Bound): **40.59%**.

Even under cross-domain transfer, PRISM retains over 60% of the in-domain adaptation gain, demonstrating that the learned spectral-manifold mapping captures domain-invariant visual-complexity priors.

Continual Knowledge Retention. We further investigated whether dynamic parameter slicing causes catastrophic forgetting of pre-trained knowledge. Using an anchor retention split from pre-training data:

- Main task adaptation accuracy improved from 50.43% to **52.17%** (+1.74 pp);
- Retained task accuracy improved from 60.53% to **76.32%** (**+15.79 pp**);
- Knowledge Retention Ratio: $\mathbf{1.2609} > 1.0$, demonstrating backward knowledge transfer rather than forgetting.

E.4 Broader Architectural Scales and General Vision-Language Benchmarks

To verify architecture-agnostic generalization, we expanded evaluations to other model generations, parameter scales, and standardized visual reasoning benchmarks:

Table 17: Cross-model scaling and extended benchmark evaluations.

Model & Scale	Evaluation Benchmark	Base Model Acc (%)	+ PRISM Acc (%)
Qwen2-VL (2B)	RealworldQA	54.50	**63.50** (+9.00 pp)
Qwen3-VL (8B)	RealworldQA	67.20	**73.00** (+5.80 pp)
Standardized Benchmarks	**Qwen3-VL-2B Base**	**+ PRISM (Ours)**	**Gemini 3 Pro Reference**
MME (Perception)	82.73	**84.42** (+1.69 pp)	85.00
POPE (Hallucination Resistance)	88.33	**90.57** (+2.24 pp)	88.00
TextVQA (OCR & Document Reasoning)	79.56	**83.59** (+4.03 pp)	93.10

As shown in Table 17, PRISM achieves substantial gains on both Qwen2 and Qwen3 architectures across 2B and 8B scales. On general benchmarks, PRISM boosts POPE hallucination resistance to **90.57%** (surpassing Gemini 3 Pro's 88.00%) and TextVQA to **83.59%**, demonstrating broad domain utility.

E.5 End-to-End Differentiable Router Training and Soft-Mask Mechanics

To resolve the discrepancy between discrete rank slicing at inference and gradient-based backpropagation during training, PRISM utilizes a **Differentiable Soft Prefix Mask**:

Continuous Relaxation during Training. The router network outputs a continuous scalar $\hat{r} \in [1, R_{max}]$ representing the predicted effective rank capacity. During forward propagation, instead of abruptly severing tensor dimensions, we apply a smooth sigmoid mask $M(\hat{r}) \in \mathbb{R}^{R_{max} \times R_{max}}$ to the master adaptation tensor:

$$M_{jk}(\hat{r}) = \mathbb{I}(j = k) \cdot \sigma\left(\frac{\hat{r} - j + 0.5}{\tau}\right), \quad j, k \in \{1, \dots, R_{max}\}, \tag{66}$$

where $\tau > 0$ denotes a temperature hyperparameter. The active update tensor during training is given by:

$$\mathcal{T}_{active} = M(\hat{r})\mathcal{T}. \tag{67}$$

The task loss gradient $\nabla_{\mathcal{T}_{active}}\mathcal{L}$ flows smoothly through $M(\hat{r})$ directly to the continuous prediction $\hat{r}$, and subsequently updates the parameters of the GLA module and the MLP router via standard chain-rule backpropagation:

$$\frac{\partial \mathcal{L}}{\partial \hat{r}} = \sum_{j=1}^{R_{max}} \frac{\partial \mathcal{L}}{\partial (\mathcal{T}_{active})_j} \cdot \mathcal{T}_j \cdot \frac{1}{\tau}\sigma\left(\frac{\hat{r} - j + 0.5}{\tau}\right)\left[1 - \sigma\left(\frac{\hat{r} - j + 0.5}{\tau}\right)\right]. \tag{68}$$

The joint training objective combines task loss with spectral regularization and a capacity budget loss:

$$\mathcal{L}_{total} = \mathcal{L}_{task} + \lambda_{spec}\mathcal{L}_{spec} + \lambda_{budget}\left(\frac{\hat{r}}{R_{max}}\right)^2. \tag{69}$$

Hard Slicing at Inference Time. At deployment, the temperature-dependent soft mask is discarded entirely. The model computes the discrete rank $r = \text{round}(\hat{r}) \in \{1, \dots, R_{max}\}$ and executes physical zero-copy sub-tensor slicing:

$$\Delta W_r = \frac{\alpha}{\sqrt{r}}\left(\mathcal{T}_B[:, 1:r] \cdot \mathcal{T}_A[1:r, :]\right). \tag{70}$$

This formulation preserves strict mathematical continuity during gradient optimization while delivering pure, unpadded sub-matrix multiplications during physical execution on edge hardware.