2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Adaptive Capacity Allocation for Vision Language Action Fine-tuning

Donghoon Kim[1], Minji Bae[1†], Unghui Nam[1†], Gyeonghun Kim[1†], Suyun Lee[1†], Kyuhong Shim[2‡], Byonghyo Shim[1‡]

***Abstract*— Vision language action models (VLAs) are increasingly used for Physical AI, but deploying a pre-trained VLA model to unseen environments, embodiments, or tasks still requires adaptation. Parameter-efficient fine-tuning (PEFT), especially LoRA, is common for VLA policies, yet the exposed capacity knob, the rank, does not transfer uniformly: robotics transfer exhibits a higher and task-varying intrinsic rank than language fine-tuning. Small ranks suffice for LLMs (e.g., $r \in \{4, 8\}$), while spectral analyses indicate VLAs may require much larger ranks (e.g., $r \approx 128$) or near–full rank, a mismatch that worsens in multi-task settings. We present LoRA-SP (Select–Prune), a rank-adaptive fine-tuning method that replaces fixed-rank updates with input- and layer-wise capacity. LoRA-SP uses an SVD-style parameterization with a small router whose nonnegative scores act as singular values over a shared vector bank. The active set is chosen by an energy target on the cumulative squared scores $E(k) \geq \eta$, providing a direct link to approximation error via our spectral analysis. During training, η concentrates energy on a few directions and teaches the router to rely on fewer vectors while preserving accuracy. This yields compact adapters that reduce cross-task interference and improve generalization. On four real-robot manipulation tasks collected on an unseen AgileX PiPER arm, across two VLA backbones (π_0 and SmolVLA), LoRA-SP matches or exceeds full fine-tuning with far fewer trainable parameters, and improves multi-task success by up to 31.6% over standard LoRA while remaining robust to rank choice.**

I. INTRODUCTION

In recent years, large multimodal models (LMMs) have been applied in a wide range of domains long considered exclusive to human intelligence, such as complex reasoning, tool use, and code generation [1], [2], [3]. Building on this progress, LMMs are moving beyond image–text perception to physical interaction with the world [4], [5], [6], [7]. This shift has given rise to physical AI: building agents that act and learn through embodied interaction in real-world settings.

Conventionally, approaches based on reinforcement learning (RL) and imitation learning (IL) have been widely used to build embodied intelligence, but these approaches are often limited to specific tasks or environments [8], [9], [10]. Recent progress has been driven by the training of LMMs with vision-language-action-paired data, enabling agents to learn generalizable mappings from visual perception and language instruction to action (a.k.a., vision-language-action (VLA) model) [11], [12]. This paradigm allows a single model to operate in diverse environments, embodiments, and task distributions, marking a significant step toward versatile embodied intelligence.

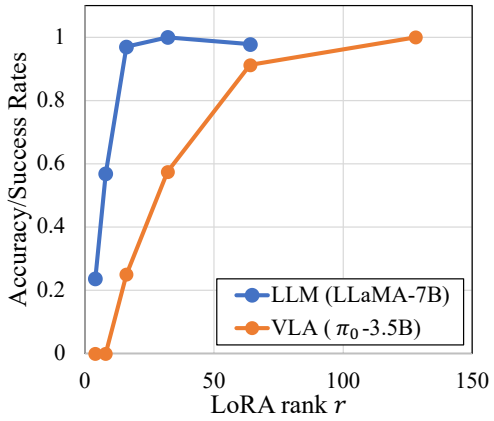


Fig. 1: Rank–performance curves (accuracy/success relative to full fine-tuning; 1.0 = full FT). LLM (LLaMA-7B) reaches near-full-FT performance with very small ranks ($r \in \{4, 8\}$), whereas VLA (π_0-3.5B) improves steadily and only approaches parity around $r \approx 128$, consistent with a higher intrinsic dimension in the VLA transfer setting.

Although Vision-Language-Action (VLA) models have made significant strides, the idea of a single universal model remains challenging. This is especially true when the robot's embodiment, task, or environment differs from those in the training data. As shown in Fig. 2, differences in hardware specifications (e.g., DoF, link lengths, and joint limits) change the feasible inverse kinematics set, so identical goals map to different joint-space solutions and thus different trajectories. Shifts in perception–action alignment, driven by camera intrinsics/extrinsics, viewpoint, and workspace scale, also change how pixel displacements translate into physical motion in the robot frame. In summary, these real-world factors modify the distribution that the policy needs to cover, thereby increasing the adaptation capacity required.

In this adaptation task, LoRA and its variants have been widely used due to their competitive performance and high parameter efficiency [13], [14], [15]. However, the rank hyperparameter, which is key to their capacity, does not

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (2022M3C1A3099336) and in part by Samsung Electronics Co., Ltd (IO251211-14335-01).

[1]D.Kim, M.Bae, U.Nam, G.Kim, S.Lee, B.Shim are with Department of Electrical and Computer Engineering, Seoul National University, Seoul 08826, Republic of Korea (email: {dhkim, mjbae, uhnam, ghkim, sylee, bshim}@islab.snu.ac.kr

[2]K.Shim is with Department of Computer Science and Engineering, Sungkyunkwan University, Suwon 16419, Republic of Korea (email: khshim@skku.edu)

[†]2nd co-authors, listing order is random

[‡]Corresponding authors

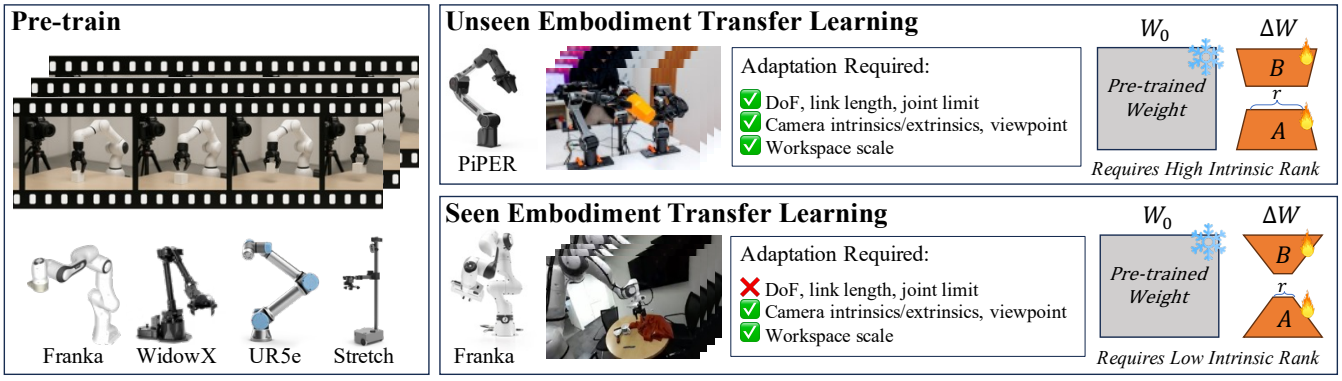


Fig. 2: VLA models are pre-trained on diverse manipulation tasks and robot embodiment (e.g., Franka Emika, WidowX, UR5e, Stretch) data. We compare transfers to a *seen* embodiment (Franka) versus an *unseen* embodiment (PiPER). Unseen-embodiment transfer changes both the robot's kinematic specification (DoF, link lengths, joint limits) and the perception geometry (camera intrinsics/extrinsics, viewpoint) and workspace scale, which pushes the update to require a *higher intrinsic rank*; by contrast, when the embodiment is seen, adaptation primarily compensates for perception/scale shifts and works with a *lower rank*.

generalize uniformly across different domains, meaning that a fixed rank might not be uniformly optimal for all cases. We observe that, for adaptation in robotics learning, the intrinsic rank (i.e., the minimal dimensionality required to capture task-relevant features) is typically higher and more variable compared to language models [16], [17]. As shown in Fig 1, while LLaMA-7B achieves near full fine-tuning performance with rank $r \in \{4,8\}$, π_0-3.5B demands ranks up to $r = 128$ to achieve the same performance [18], [12]. Even in single-task training, the optimal rank varies with task difficulty (see Fig. 3 (a)). This uncertainty about the rank makes it difficult to select a single global rank in multi-task settings (see Fig. 3 (b)). Such a choice may force heterogeneous tasks to share a fixed subspace, which increases cross-task interference (competition for the shared adapter subspace) and reduces positive transfer. In practice, we perform the brute sweeping (e.g., grid search) of the rank parameter to find a near-optimal rank for each setting. The cost of brute-force sweeps highlights the need for rank-adaptive frameworks that dynamically allocate capacity to task- and embodiment-specific demands.

To this end, we introduce LoRA-SP (Select-Prune), a rank-adaptive fine-tuning technique designed to resolve the limitation of fixed-rank LoRA approaches. Classic LoRA updates the weight W with a fixed-rank factorization $\Delta W = BA$ so the module computes $(W + \Delta W)x = Wx + B(Ax)$. We generalize this low-rank form by replacing BA with SVD-style parameterization $U\,\mathrm{diag}(s(x))\,V$ where U and V define a vector bank (basis) and the router outputs nonnegative *singular-value-like* scores $s(x) \in \mathbb{R}^r$. We start from a sufficiently large rank r (smaller than dense layers but large enough to cover necessary directions) and let the router learn, per input and per layer, which basis vectors are active and what magnitudes (singular values) they should take. We then choose the smallest active rank k such that the cumulative singular-value energy satisfies $E(k) \geq \eta$, zeroing the remaining vectors for that input. We also add a spectral loss $\mathscr{L}_{\text{spec}} = 1 - E_k(x)$ that concentrates energy onto the selected vectors, creating a progressive concentration where useful directions are amplified across iterations. As a result, LoRA-SP learns to realize tasks with only the minimal active set needed, yielding compact adapters that preserve accuracy while lowering inference cost.

We evaluate our method on four real-world manipulation tasks collected with an unseen 7-DoF AgileX PiPER arm, totaling 480 demonstrations with dual RGB views. Experiments are conducted on two pretrained VLA backbones: π_0[12] and SmolVLA[19], which represent a high-capacity and a lightweight model, respectively. These are compared against several baselines including full fine-tuning, standard LoRA with ranks $r \in \{16,32,64,128\}$, AdaLoRA [20], and LoRA-MoE [21] using top-1 and weighted-sum routing. Standard LoRA shows clear rank sensitivity, performing well in single-task settings at high ranks but collapsing under multi-task training due to mismatched task capacities and subspace interference. Neither AdaLoRA nor LoRA-MoE consistently outperform standard LoRA in the multi-task regime. In contrast, LoRA-SP achieves consistently strong multi-task performance across all tasks and backbones, while updating significantly fewer parameters than full fine-tuning (Table I, II). An ablation study shows that the spectral loss enables effective rank pruning without accuracy loss, while threshold ablation confirms LoRA-SP's robustness under reduced active ranks (Table III, IV).

Our contributions are summarized as follows:

- We quantify rank needs via cumulative energy $E(k)$ and rank–performance curves, showing that OOD embodiment transfer (e.g., AgileX PiPER) requires substantially larger ranks than language fine-tuning and exhibits strong rank sensitivity (Figs. 1, 2). This motivates rank-adaptive capacity instead of a fixed global rank.
- We introduce a fine-tuning method that adaptively adjusts trainable capacity per input and layer. Router produces singular-value–like scores $s(x)$ over a shared vector bank, and the effective rank is set by an energy target on the cumulative squared scores. Spectral con-

centration loss amplifies the surviving vectors, creating a positive feedback that progressively reduces the active set, while the task loss preserves accuracy.

- We validate LoRA-SP on four real-world manipulation tasks with 7-DoF AgileX PiPER arm, using π_0 and SmolVLA backbones. Compared to the baselines LoRA-SP achieves comparable or better performance with significantly fewer trainable parameters and activated ranks. It improves multi-task success by up to 31.6% over standard LoRA while remaining robust to rank choice (Sec. V, Tabs. I, Tabs. II,III).

II. Theoretical Background and Rationale

A. Intrinsic Dimension of Fine-tuning

We formalize the *intrinsic dimension* (ID) of a task as the smallest update capacity needed to recover a target performance (e.g., the performance achieved by full fine-tuning). Let a layer have parameters $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, training objective $L(\cdot)$, initialization W_0, and a target value $L_\star$ (e.g., the loss achieved by full fine-tuning). Under LoRA, we constrain the update to be low rank, $\Delta W = BA$, with $B \in \mathbb{R}^{d_{\text{out}} \times r}$ and $A \in \mathbb{R}^{r \times d_{\text{in}}}$, so that $\text{rank}(\Delta W) \le r$ [16], [13]. For tolerance $\varepsilon \ge 0$, we define the LoRA-based intrinsic dimension as

$$\text{ID}_\varepsilon^{\text{LoRA}} = \min \left\{ r : \begin{array}{l} \exists A \in \mathbb{R}^{r \times d_{\text{in}}},\ B \in \mathbb{R}^{d_{\text{out}} \times r} \\ \text{s.t. } L\big(W_0 + BA\big) \le L_\star + \varepsilon \end{array} \right\}. \quad (1)$$

In words, instead of optimizing all entries of W, we restrict the update to a rank-r factorization BA. If such (A,B) achieve the target loss within ε, then r is feasible; $\text{ID}_\varepsilon^{\text{LoRA}}$ is the minimum feasible rank. This yields an operational measure of the task's effective degrees of freedom that is largely decoupled from the ambient parameter count.

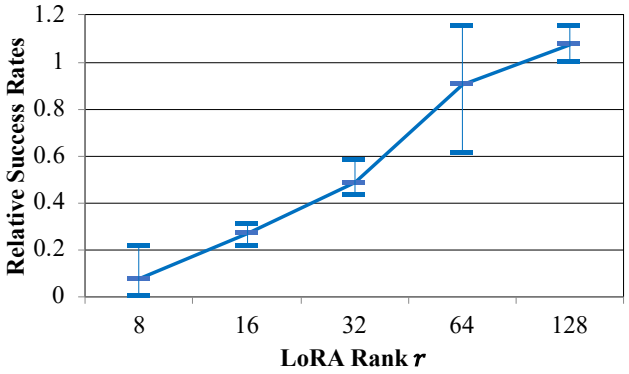
Building on this structure, subsequent methods tie rank selection to data- and layer-specific sensitivity using spectral diagnostics. For example, AdaLoRA allocates a global rank budget across layers using SVD-based importance scores, placing the capacity where it is needed most [20].

B. LoRA-MoE and Multi-task Performance

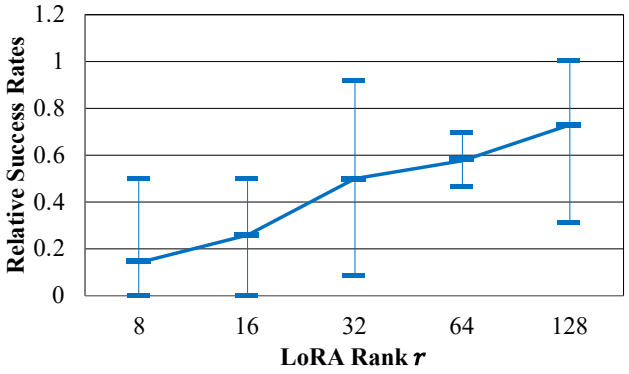
A common extension of LoRA to multi-task settings is to view each task as a small low-rank perturbation applied to a shared base policy. Let $f(x;\theta)$ be the frozen base model and, for each task i (e.g., pick-and-place, tool use), let $\Delta\theta_i$ be a LoRA update such that the single-task policy $f_i(x) = f(x;\theta+\Delta\theta_i)$ performs well on task i. In a pure single-task fine-tune we would deploy $\theta+\Delta\theta_i$ for that task; in the multi-task case we instead add a routing function $g(x)$ that selects or mixes these adapters at inference, yielding

$$\tilde{f}(x) = f\left(x;\, \theta + \sum_i g_i(x)\,\Delta\theta_i\right), \quad (2)$$

where $g(x)$ can be a hard selector ($g_i \in \{0,1\}$, one expert per input) or a soft mixture ($\sum_i g_i = 1$) [21]. Under ideal routing and disjoint task distributions, $\tilde{f}$ matches each single-task expert on its own data. In practice, however, capacity scales with the number of experts, increasing memory and latency; performance is also sensitive to routing errors; and useful update directions remain siloed within task-specific adapters, limiting sharing. Moreover, both the LoRA rank per expert and the number of experts are sensitive hyperparameters that typically require dataset-specific sweeps. These factors make LoRA-MoE hard to tune and deploy at scale; by contrast, LoRA-SP uses a single adapter with input- and layer-conditioned capacity, avoiding expert proliferation while enabling parameter sharing (see Sec. IV).



(a) Single Task Fine-tuning



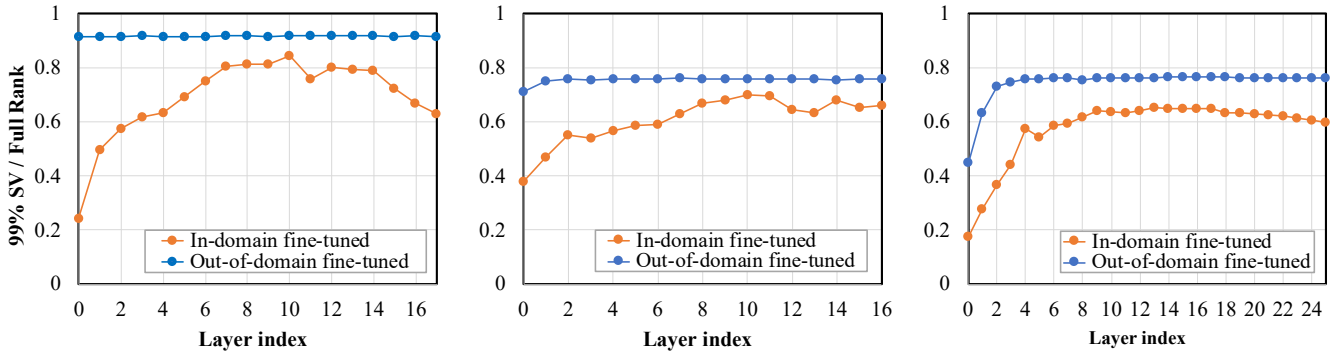
(b) Multi Task Fine-tuning

Fig. 3: Rank sensitivity in single- and multi-task LoRA fine-tuning on π_0 model. (a) LoRA modules trained independently on each single task. While single-task modules also require higher ranks to reach full performance, their variance across tasks is lower than in the multi-task setting. Together, the results highlight the difficulty of choosing a single global rank that balances efficiency and accuracy across tasks, motivating rank-adaptive allocation. (b) Multi-task LoRA fine-tuning across four manipulation tasks. Success rate increases with rank but exhibits substantial variance across tasks, reflecting interference and heterogeneous capacity needs.

III. Analysis: Intrinsic Dimension and Spectral Error

A. Spectral Error in Low Rank Approximation of Gradients

Given a rank r on a weight update, *the smallest attainable relative Frobenius error is determined by the spectrum*: if $E(k)$ denotes the cumulative energy of the top-k singular values (Eq. 3), then the *best* rank-k approximation achieves



(a) Gemma Action Expert (b) Language Model (c) Vision Tower

Fig. 4: **Spectral Rank Variation by Embodiment During π_0 Fine-tuning.** Number of singular values required to capture 99% of the total energy (normalized by the full rank) across different layers and modules. We compare π_0 models fine-tuned on in-domain and out-of-domain data. The in-domain model is fine-tuned on the DROID dataset, which uses the robotic arm (Franka Panda) included in π_0's pretraining data. The out-of-domain model is fine-tuned on a dataset collected with the AgileX PiPER robotic arm, an embodiment absent from the pretraining data. The results show that the required rank varies by embodiment, and generalizing to a novel embodiment demands higher-rank to achieve comparable performance.

error $\sqrt{1-E(k)}$ (Eq. 4). Thus, *choosing ranks* in LoRA is equivalent to *controlling* these spectral error.

Definition: Let A denote a weight or update matrix. Write its singular values as $\sigma_1 \geq \cdots \geq \sigma_r > 0$ with $r = \mathrm{rank}(A)$, and define the cumulative energy

$$E(k) = \frac{\sum_{i=1}^{k} \sigma_i^2}{\sum_{i=1}^{r} \sigma_i^2}. \tag{3}$$

If A_k is the rank-k truncated SVD of A, then we have the following identity and optimality statement.

Proposition: Let $A = U\Sigma V^\top$ be an SVD parametrization with $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_r, 0, \ldots)$ and let $\Sigma_k = \mathrm{diag}(\sigma_1, \ldots, \sigma_k, 0, \ldots)$ so that $A_k = U\Sigma_k V^\top$. Then

$$\frac{\|A - A_k\|_F}{\|A\|_F} = \sqrt{1 - E(k)}, \tag{4}$$

equivalently,

$$\|A - A_k\|_F^2 = \sum_{i=k+1}^{r} \sigma_i^2. \tag{5}$$

Moreover, A_k is a best rank-k approximation to A in Frobenius norm, i.e., $\|A - B\|_F \geq \|A - A_k\|_F$ for every matrix B with $\mathrm{rank}(B) \leq k$.

Proof: By orthogonal invariance of the Frobenius norm,

$$\|A - A_k\|_F = \|U(\Sigma - \Sigma_k)V^\top\|_F = \|\Sigma - \Sigma_k\|_F. \tag{6}$$

Since $\Sigma - \Sigma_k = \mathrm{diag}(0, \ldots, 0, \sigma_{k+1}, \ldots, \sigma_r)$, we obtain

$$\|A - A_k\|_F^2 = \|\Sigma - \Sigma_k\|_F^2 = \sum_{i=k+1}^{r} \sigma_i^2. \tag{7}$$

Also,

$$\|A\|_F^2 = \|\Sigma\|_F^2 = \sum_{i=1}^{r} \sigma_i^2. \tag{8}$$

Dividing the two identities yields Eq. (4) with $E(k)$ as in Eq. (3). The optimality of A_k among all rank-k matrices follows from the Eckart–Young–Mirsky theorem. □

B. Characteristics of Gradients in VLA Models

Intrinsic Dimension As shown in the rank–performance curves of Fig. 1, while the language model (LLaMA-7B) approaches full fine-tuning (FT) performance even at low ranks ($r \in \{4, 8\}$), the vision–language–action (VLA) model π_0-3.5B requires ranks up to $r \approx 128$ to achieve full FT performance. This suggests that the intrinsic dimension of the gradient in VLA tasks is significantly higher, necessitating more expressive update directions for effective adaptation. Supporting evidence from the spectral energy curves in Fig. 4 reveals that the minimum rank k satisfying $E(k) \geq 0.99$ (based on the relative error bound in Eq. (4)) varies widely between modules and downstream dataset, ranging from 0.2 to 0.9 of the full rank in the Gemma expert, language model, and vision tower. This wide distribution of required ranks across layers and domains clearly exposes the limitations of uniform rank allocation.

Rank Sensitivity Fine-tuning π_0 in out-of-domain settings (e.g., robotic arms not seen during pre-training) further increases the effective dimension of gradients. For instance, Fig. 4 shows that datasets collected with the AgileX PiPER robotic arm (out-of-domain) consistently require higher spectral ranks across all modules compared to in-domain data from the Franka Panda. Similar patterns emerge in multi-task learning (Fig. 3): while single-task modules also demand high ranks, their performance variance across tasks is lower than in multi-task settings. When fine-tuning across four manipulation tasks, substantial performance variance arises due to task interference and heterogeneous capacity demands. This highlights the difficulty of balancing efficiency and accuracy with a single global rank, reinforcing the need for adaptive rank allocation.

IV. LoRA Select–Prune (LoRA-SP)

A. Overview

LoRA-SP replaces a single fixed rank with data-conditioned capacity that varies by input and layer. As shown

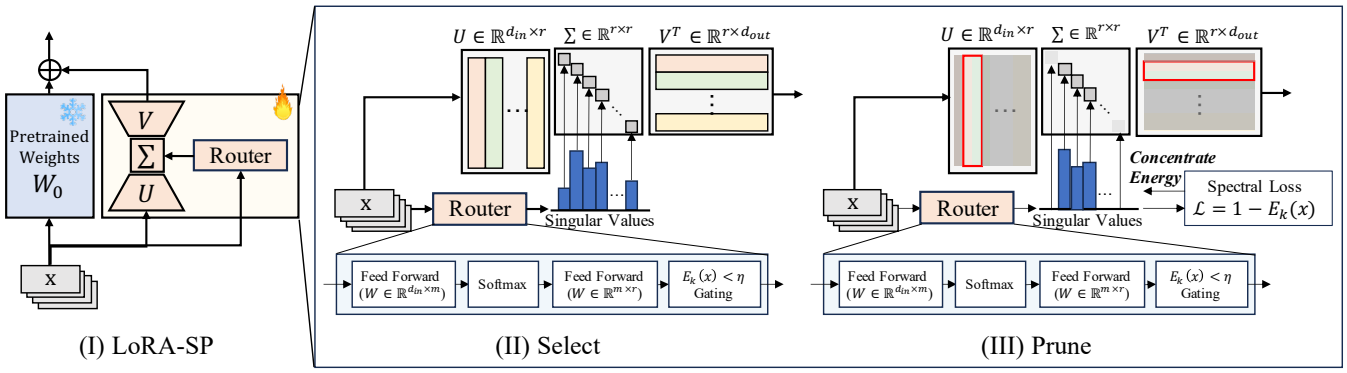


Fig. 5: **LoRA-SP (Select–Prune).** (I) Overview: a wide vector bank (U,V) is trained together with a router on the backbone W_0. (II) Select: the router produces vector-level scores that act as singular values, forming an input- and layer-conditioned update $\Delta W = U\,\Sigma(x)\,V$; the histogram illustrates the spectral energy distribution across vectors. (III) Prune: only the smallest set of basis vectors whose cumulative energy exceeds the target η are kept, progressively reducing the active rank while maintaining accuracy.

in Fig. 5, we train a shared vector bank (U,V) and a small router. For each input x, the router scores basis vectors and we keep only the smallest set whose cumulative score energy meets a target η. This realizes the update with a few active vectors at test time and reduces interference across tasks by reusing only the vectors that matter for the current input.

B. Problem Setup and Connection to LoRA

We adapt a pretrained backbone $f(W;x)$ with layer weights $\{W_\ell\}$ for multi-task data (x,t). Classic LoRA applies a fixed-rank update $\Delta W = BA$ so the forward map is $(W+\Delta W)x = Wx + B(Ax)$. LoRA-SP generalizes this by replacing BA with an input-conditioned SVD-style form

$$\Delta W_\ell(x) \;=\; U_\ell \,\mathrm{diag}\big(s_\ell(x)\big)\, V_\ell, \tag{9}$$

where $U_\ell \in \mathbb{R}^{d_{\text{out}} \times r}$ and $V_\ell \in \mathbb{R}^{r \times d_{\text{in}}}$ define a per-layer vector bank, and $s_\ell(x) \in \mathbb{R}^r_{\geq 0}$ indicates router scores. Intuitively, $s_\ell(x)$ plays the role of data-conditioned singular values: large scores mark directions that should be active for the current input.

C. Select: Vector-level Gating

Wide Initialization We initialize a wide vector bank U and V per module (r=128 in all experiments) and train it jointly with a lightweight two-layer router (Fig. 5 (II)). Although full rank would be ideal in principle, before adaptation we do not know which directions in the representation space will be useful for a given input or layer. A wide initialization therefore provides sufficient coverage of candidate directions without committing to a specific subspace; the router and the spectral loss then discover a compact, task-relevant subspace and deactivate the rest.

Singular Value Generation Unlike module-level MoE (routes to one expert per call), LoRA-SP gates at the vector level, so the number of active vectors (i.e., LoRA rank) is adjusted per input and per layer. Given input $x \in \mathbb{R}^{d_{\text{in}}}$, the router produces singular value-like scores $s(x)$ through a two-layer MLP with activation ϕ:

$$h_1(x) = \phi(W_1 x + b_1), \qquad s(x) = W_2 h_1(x) + b_2 \in \mathbb{R}^r_{\geq 0}. \tag{10}$$

With $U \in \mathbb{R}^{d_{\text{out}} \times r}$ and $V \in \mathbb{R}^{r \times d_{\text{in}}}$, the update becomes

$$\Sigma(x) = \mathrm{diag}\big(s(x)\big), \qquad \Delta W(x) = U\,\Sigma(x)\,V. \tag{11}$$

This construction lets LoRA-SP flexibly adjust both the rank and the direction of updates.

Active Vector Selection Following Sec. III, we determine the active rank by controlling the Frobenius error of low-rank approximation. Specifically, we sort generated singular values by $s_i(x)^2$ and define the cumulative energy

$$E_k(x) = \frac{\sum_{i=1}^{k} s_i(x)^2}{\sum_{j=1}^{r} s_j(x)^2}. \tag{12}$$

The effective rank k is chosen as the smallest index satisfying $E_k(x) \geq \eta$, and the singular values beyond k are zeroed. Because $E_k(x)$ bounds the relative approximation error as $\sqrt{1-E_k(x)}$ (Eq. 4), η serves as an explicit tolerance knob (e.g., $\eta = 0.99$ implies ≤ 0.1 error). This procedure guarantees that updates retain task-relevant directions while automatically discarding low-energy vectors, yielding efficient and input-specific capacity allocation.

D. Prune: Active-set Reduction via Spectral Loss

Spectral Loss We add a spectral loss term which encourages the router to concentrate energy onto the vectors that survive the η-based gating:

$$\mathscr{L}_{\text{spec}}(x) = 1 - E_k(x). \tag{13}$$

This creates a reinforcement loop: once a vector is selected, $\mathscr{L}_{\text{spec}}$ pushes its singular value higher, making it even more likely to be selected again. Over training, singular-value mass is gradually shifted toward a small stable set of directions, while the task loss prevents collapse to trivial solutions. Empirically, we observe that task success rates remain stable even when the number of active vectors is nearly halved, showing that pruning through spectral concentration yields compact adapters with minimal accuracy loss.

TABLE I: Trainable parameters, active ranks, and multi-task success rates across various fine-tuning strategies. Active rank denotes effective rank k per token in average across all layers.

Model	Strategy	Trainable / Total (%)	Active Rank	Multi-task Success Rate (%)			
				Open	Pour	Press	Pick-Place
π_0 [12]	LoRA ($r = 128$) [13]	9.1	128	73.3	26.7	80.0	60.0
	LoRA-MoE (top-1) [21]	9.2	32	13.3	13.3	53.3	13.3
	LoRA-MoE (weighted sum) [21]	9.2	128	46.7	60.0	93.3	80.0
	AdaLoRA [20]	9.1	76	20.0	6.7	40.0	60.0
	Full FT	100.0	Full	80.0	86.7	80.0	86.7
	LoRA-SP	9.2	76	80.0	80.0	93.3	80.0
SmolVLA [19]	LoRA ($r = 128$) [13]	17.0	128	40.0	20.0	93.3	86.7
	LoRA-MoE (top-1) [21]	17.2	32	33.3	46.7	86.7	73.3
	LoRA-MoE (weighted sum) [21]	17.2	128	60.0	80.0	100.0	66.7
	AdaLoRA [20]	17.0	60	6.7	0.0	40.0	20.0
	Full FT	100.0	Full	73.3	86.7	100.0	86.7
	LoRA-SP	17.1	60	86.7	86.7	100.0	93.3

TABLE II: Comparison of task success rates across varying LoRA ranks under single-task and multi-task training.

Model	Strategy	Trainable / Total (%)	Active Rank	Success Rate (%)							
				Single-task Training				Multi-task Training			
				Open	Pour	Press	Pick-Place	Open	Pour	Press	Pick-Place
π_0 [12]	LoRA [13]	0.6	8	6.7	0.0	0.0	20.0	0.0	6.7	40.0	0.0
		1.2	16	26.7	20.0	26.7	20.0	0.0	20.0	40.0	26.7
		2.4	32	40.0	46.7	40.0	40.0	6.7	53.3	73.3	33.3
		4.8	64	53.3	73.3	100.0	86.7	46.7	40.0	46.7	60.0
		9.1	128	93.3	80.0	100.0	100.0	73.3	26.7	80.0	60.0
	Full FT	100.0	Full	86.7	80.0	86.7	93.3	80.0	86.7	80.0	86.7
	LoRA-SP	9.2	76	73.3	80.0	80.0	80.0	60.0	80.0	93.3	80.0
SmolVLA [19]	LoRA [13]	1.3	8	0.0	6.7	26.7	20.0	0.0	0.0	26.7	0.0
		2.5	16	53.3	60.0	80.0	60.0	0.0	13.3	86.7	0.0
		4.9	32	60.0	93.3	93.3	80.0	13.3	0.0	86.7	26.7
		9.3	64	73.3	86.7	100.0	80.0	26.7	26.7	80.0	86.7
		17.0	128	86.7	80.0	100.0	100.0	40.0	20.0	93.3	86.7
	Full FT	100.0	Full	86.7	86.7	100.0	100.0	73.3	86.7	100.0	86.7
	LoRA-SP	17.1	60	86.7	93.3	93.3	100.0	86.7	86.7	100.0	93.3

E. Training Losses

The overall training loss combines task optimization, spectral concentration, and router regularization:

$$\mathscr{L} = \mathbb{E}[\mathscr{L}_{\text{task}}] + 10^{-2}\,\mathbb{E}[\mathscr{L}_{\text{spec}}] + 10^{-3}\,\mathbb{E}[\mathscr{L}_{\text{router}}], \quad (14)$$

where $\mathscr{L}_{\text{task}}$ is the main objective (e.g., flow matching), and $\mathscr{L}_{\text{router}}$ includes balance [22] and z-loss [23] terms.

V. EXPERIMENTS

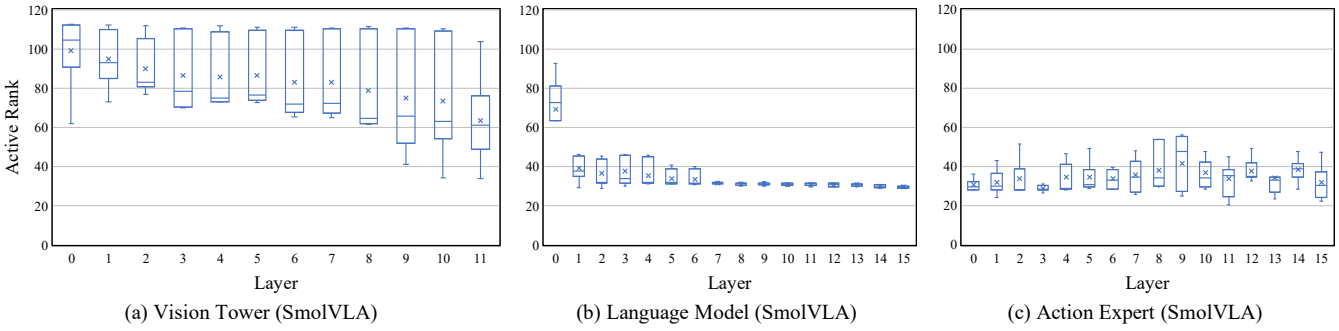
A. Experiments Setup

Baselines We evaluate our method against several widely used and recent fine-tuning strategies, including full fine-tuning (Full FT), standard low-rank adaptation (LoRA), and expressive variants such as LoRA-MoE (top-1 and weighted-sum routing) and AdaLoRA. All methods are trained under identical conditions to ensure fair comparison and our method, LoRA-SP, is trained with an initial rank of 128 and the energy target $\eta = 0.9$. These strategies are applied to two pretrained VLA backbones: π_0, a large-capacity policy based on PaLIGemma [24], and SmolVLA, an efficient variant built on SmolVLM-2 [25]. Performance is evaluated in terms of task success rate under single-task and multi-task training regimes.

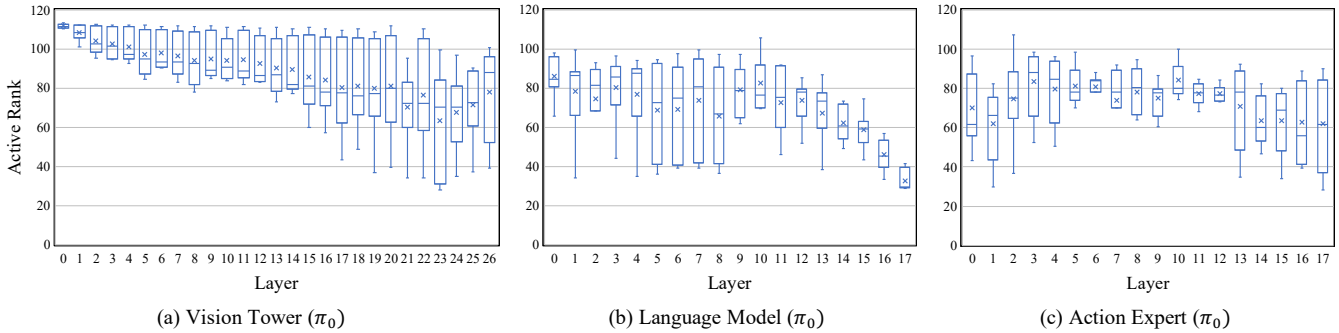
Tasks Fig. 7 shows the real-world setup for the four manipulation tasks used in our experiments: Open the Pot (Open), Pour the Block into the Basket (Pour), Press the Button (Press), and Pick and Place the Grape in the Basket (Pick-Place). These tasks cover diverse object interactions and assess the model's generalization and adaptation.

Robot Embodiment All experiments were conducted using the AgileX PiPER robotic arm, a 7-degree-of-freedom (7-DoF) manipulator. This robot is not included in most existing VLA datasets used for pretraining large models, which makes it a suitable testbed for evaluating generalization to unseen embodiments.

Datasets We collected real-world demonstration data using human teleoperation. For each task, we captured 120 episodes, resulting in a total of 480 demonstrations. Each episode was collected using two RGB camera views: a side-



(a) Vision Tower (SmolVLA) (b) Language Model (SmolVLA) (c) Action Expert (SmolVLA)



(a) Vision Tower (π_0) (b) Language Model (π_0) (c) Action Expert (π_0)

Fig. 6: Layer-wise distributions of active rank learned by LoRA-SP on validation data. Results are shown for (a) vision tower, (b) language model, and (c) action expert, for both SmolVLA (top) and π_0 (bottom) backbones. Each boxplot shows min–LQ–median–UQ–max of the active rank across inputs. The vision tower consistently requires the highest ranks, the action expert shows wide variability, while the language model layers remain comparatively low and stable. This pattern highlights strong heterogeneity across modules and underscores the limitation of using a single global rank for VLA adaptation.

(a) Open the Pot (b) Pour the Block into the Basket

(c) Press the Button (d) Pick and Place the Grape in the Basket

Fig. 7: Real-world experimental setup for four tasks: (a) Open the Pot, (b) Pour the Block into the Basket, (c) Press the Button, and (d) Pick and Place the Grape in the Basket.

view camera and a wrist-mounted camera on the robot arm.

B. Main Results

We compare adaptation methods under the multi-task training setting (Table I). Full fine-tuning yields the highest success rates but comes with high computational cost during training. Standard LoRA shows degraded performance even at rank 128, particularly on Pour. LoRA-MoE and AdaLoRA also fail to consistently outperform standard LoRA. LoRA-MoE lacks fine-grained rank control due to its expert-level gating, while AdaLoRA relies on LLM-based importance scores that misalign with VLA-specific adaptation needs. Conversely, LoRA-SP sustains uniformly high performance across tasks while modifying relatively few parameters. It improves average success rates over standard LoRA by 23.3% on π_0 and 31.6% on SmolVLA, while often matching the performance of full fine-tuning.

Table II further compares LoRA across various ranks under both single- and multi-task settings. While the success rate in single-task training improves with the rank, that in

TABLE III: Ablation study on the spectral loss in LoRA-SP.

	Active Rank (V, L, A)	Multi-task Success Rate (%)			
		Open	Pour	Press	Pick-Place
w/o Spectral	83, 107, 57	73.3	66.7	100.0	73.3
LoRA-SP	84, 35, 34	86.7	86.7	100.0	93.3

TABLE IV: Ablation study on the energy target η.

η	Active Rank	Multi-task Success Rate (%)			
		Open	Pour	Press	Pick-Place
0.5	30	6.7	13.3	93.3	13.3
0.7	46	53.3	80.0	100.0	53.3
0.8	56	80.0	86.7	100.0	60.0
0.9	60	86.7	86.7	100.0	93.3
0.99	114	80.0	86.7	100.0	100.0

multi-task training collapses regardless of ranks. This is due to two factors: (1) the optimal rank differs by task in terms of its success rate saturation, showing the inadequacy of a single global rank; (2) LoRA lacks task-specific separation, sharing the same low-rank directions across tasks, leading to interference and degraded performance.

To analyze how LoRA-SP allocates capacity, we examine its layer-wise rank distribution across model components. Fig. 6 shows that the vision module requires consistently high-rank updates, while the language and action modules remain low-rank. This highlights a key limitation of fixed-rank methods, which assign uniform capacity regardless of modules. In contrast, LoRA-SP adaptively concentrates rank in high-demand modules and prunes others, preserving performance while reducing trainable parameters.

C. Ablation Study

Spectral Loss This ablation compares multi-task performance and module-wise active rank (vision tower, language model, and action expert) with and without the spectral loss. As shown in Table III, removing the spectral loss significantly increases active rank, especially in the language module where the active rank rises from 35 to 107. At the same time, task success rates drop across several tasks. The spectral loss guides the model to retain only the most salient rank components, preventing task-irrelevant activations that lead to interference and hinder task success. It preserves higher capacity in the vision tower while pruning redundant ranks in the language and action modules, resulting in improved efficiency without compromising performance.

Energy target Table IV reports an ablation on the energy target η, which controls the cumulative singular-value energy required to activate rank k. As η decreases, the active rank decreases, reducing the number of basis vectors used in inference. We observe a clear trade-off: very low targets (e.g., $\eta = 0.5$) severely underfit most tasks, while moderate targets ($\eta = 0.7$, 0.8) achieve strong multi-task performance with substantially fewer active vectors. Also, performance saturates around $\eta = 0.9$, and setting $\eta = 0.99$ nearly doubles the effective rank with marginal additional gains. This validates our design: the energy target η directly tunes the accuracy–efficiency balance, and LoRA-SP maintains high success rates even at reduced ranks.

VI. Conclusion

We introduced LoRA-SP, a rank-adaptive fine-tuning method for VLA models. Instead of using one fixed rank, LoRA-SP assigns capacity per input and per layer. It uses an SVD-style update with vector-level gating, an energy target on the cumulative squared scores. This focuses the update on a few useful directions and prunes the rest, yielding compact adapters with robust rank choice and less cross-task interference. On four real-robot tasks with two VLA backbones, LoRA-SP matches or exceeds full fine-tuning with far fewer trainable parameters and a smaller active rank, and improves multi-task success by up to 31.6%.

References

[1] OpenAI, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023. [Online]. Available: https://arxiv.org/abs/2303.08774

[2] Gemini Team *et al.*, "Gemini: A family of highly capable multimodal models," *arXiv preprint arXiv:2312.11805*, 2023. [Online]. Available: https://arxiv.org/abs/2312.11805

[3] H. Liu *et al.*, "Visual instruction tuning," *Advances in neural information processing systems*, vol. 36, pp. 34 892–34 916, 2023.

[4] A. Brohan *et al.*, "Rt-1: Robotics transformer for real-world control at scale," *arXiv preprint arXiv:2212.06817*, 2022.

[5] B. Zitkovich *et al.*, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," pp. 2165–2183, 2023.

[6] M. J. Kim *et al.*, "Openvla: An open-source vision-language-action model," *arXiv preprint arXiv:2406.09246*, 2024.

[7] G. DeepMind, "Gemini robotics: Bringing ai into the physical world," *arXiv preprint arXiv:2503.20020*, 2025.

[8] S. Levine *et al.*, "End-to-end training of deep visuomotor policies," *Journal of Machine Learning Research*, vol. 17, no. 39, pp. 1–40, 2016. [Online]. Available: https://jmlr.org/papers/v17/15-522.html

[9] A. Rajeswaran *et al.*, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations," in *Robotics: Science and Systems (RSS)*, 2018. [Online]. Available: https://arxiv.org/abs/1709.10087

[10] D. Kalashnikov *et al.*, "Scalable deep reinforcement learning for vision-based robotic manipulation," in *Proceedings of the 2nd Conference on Robot Learning (CoRL)*, ser. Proceedings of Machine Learning Research, vol. 87, 2018. [Online]. Available: https://proceedings.mlr.press/v87/kalashnikov18a/kalashnikov18a.pdf

[11] A. O'Neill *et al.*, "Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 6892–6903.

[12] K. Black *et al.*, "
pi_0: A vision-language-action flow model for general robot control," 2024. [Online]. Available: https://arxiv.org/abs/2410.24164

[13] E. J. Hu *et al.*, "Lora: Low-rank adaptation of large language models," in *The Tenth International Conference on Learning Representations (ICLR 2022)*. OpenReview.net, 2022.

[14] T. Dettmers *et al.*, "Qlora: Efficient finetuning of quantized llms," *Advances in neural information processing systems*, vol. 36, pp. 10 088–10 115, 2023.

[15] S.-Y. Liu *et al.*, "Dora: Weight-decomposed low-rank adaptation," in *Forty-first International Conference on Machine Learning*.

[16] C. Li *et al.*, "Measuring the intrinsic dimension of objective landscapes," in *6th International Conference on Learning Representations (ICLR)*, Vancouver, BC, Canada, 2018.

[17] A. Aghajanyan *et al.*, "Intrinsic dimensionality explains the effectiveness of language model fine-tuning," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Online: Association for Computational Linguistics, Aug. 2021, pp. 7319–7328.

[18] H. Touvron *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[19] M. Shukor *et al.*, "Smolvla: A vision-language-action model for affordable and efficient robotics," 2025. [Online]. Available: https://arxiv.org/abs/2506.01844

[20] Q. Zhang *et al.*, "Adaptive budget allocation for parameter-efficient fine-tuning," *CoRR*, vol. abs/2303.10512, 2023.

[21] S. Dou *et al.*, "Loramoe: Alleviating world knowledge forgetting in large language models via moe-style plugin," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024, pp. 1932–1945.

[22] W. Fedus *et al.*, "Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity," *Journal of Machine Learning Research*, vol. 23, no. 120, pp. 1–39, 2022.

[23] B. Zoph *et al.*, "St-moe: Designing stable and transferable sparse expert models," *arXiv preprint arXiv:2202.08906*, 2022.

[24] L. Beyer *et al.*, "Paligemma: A versatile 3b vlm for transfer," 2024. [Online]. Available: https://arxiv.org/abs/2407.07726

[25] A. Marafioti *et al.*, "Smolvlm: Redefining small and efficient multimodal models," 2025. [Online]. Available: https://arxiv.org/abs/2504.05299