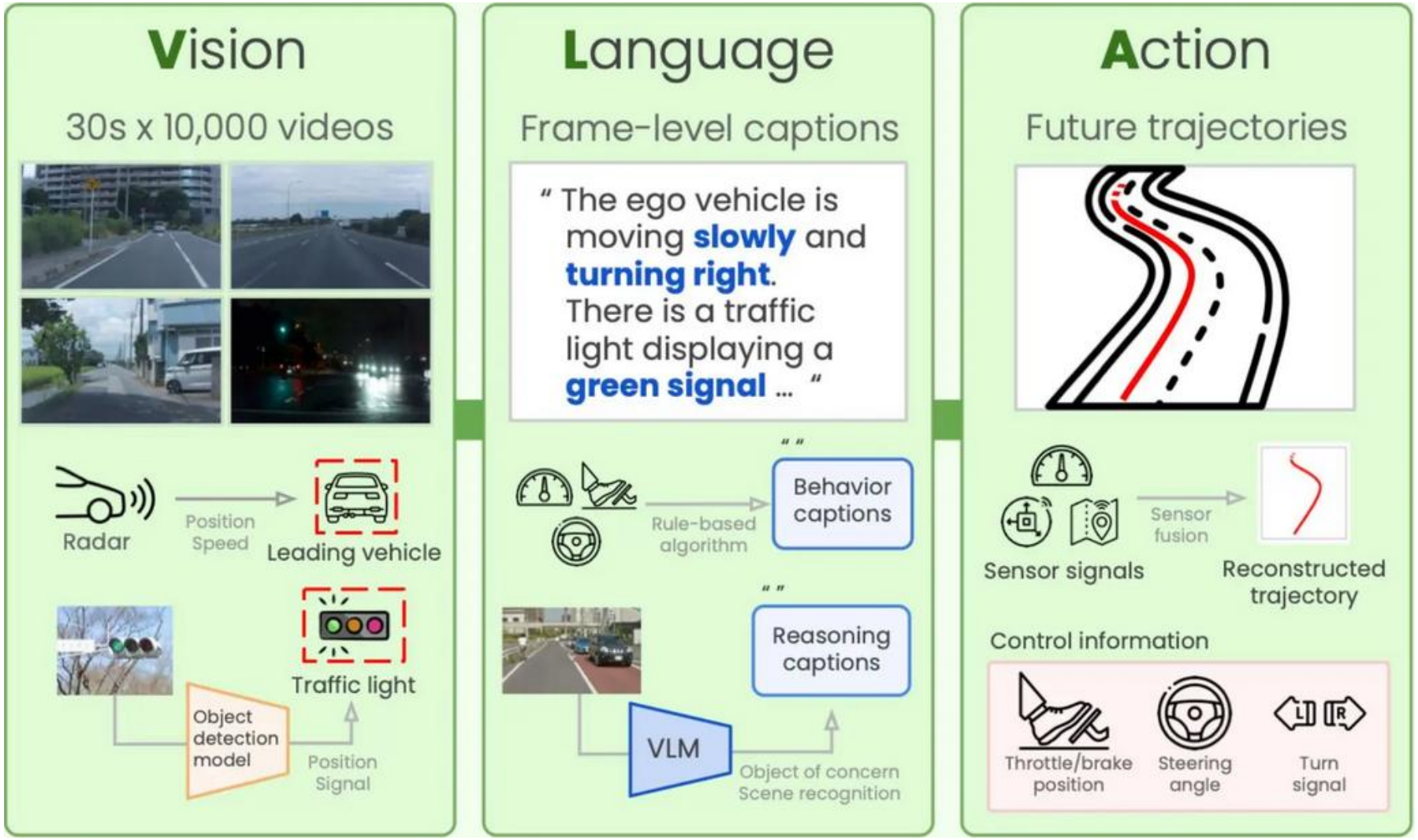
RLRC: Reinforcement Learning-based Recovery for Compressed Vision-Language-Action Models

Yuxuan Chen and Xiao Li
School of Mechanical Engineering, Shanghai Jiao Tong University

2025.08.07.
김동민

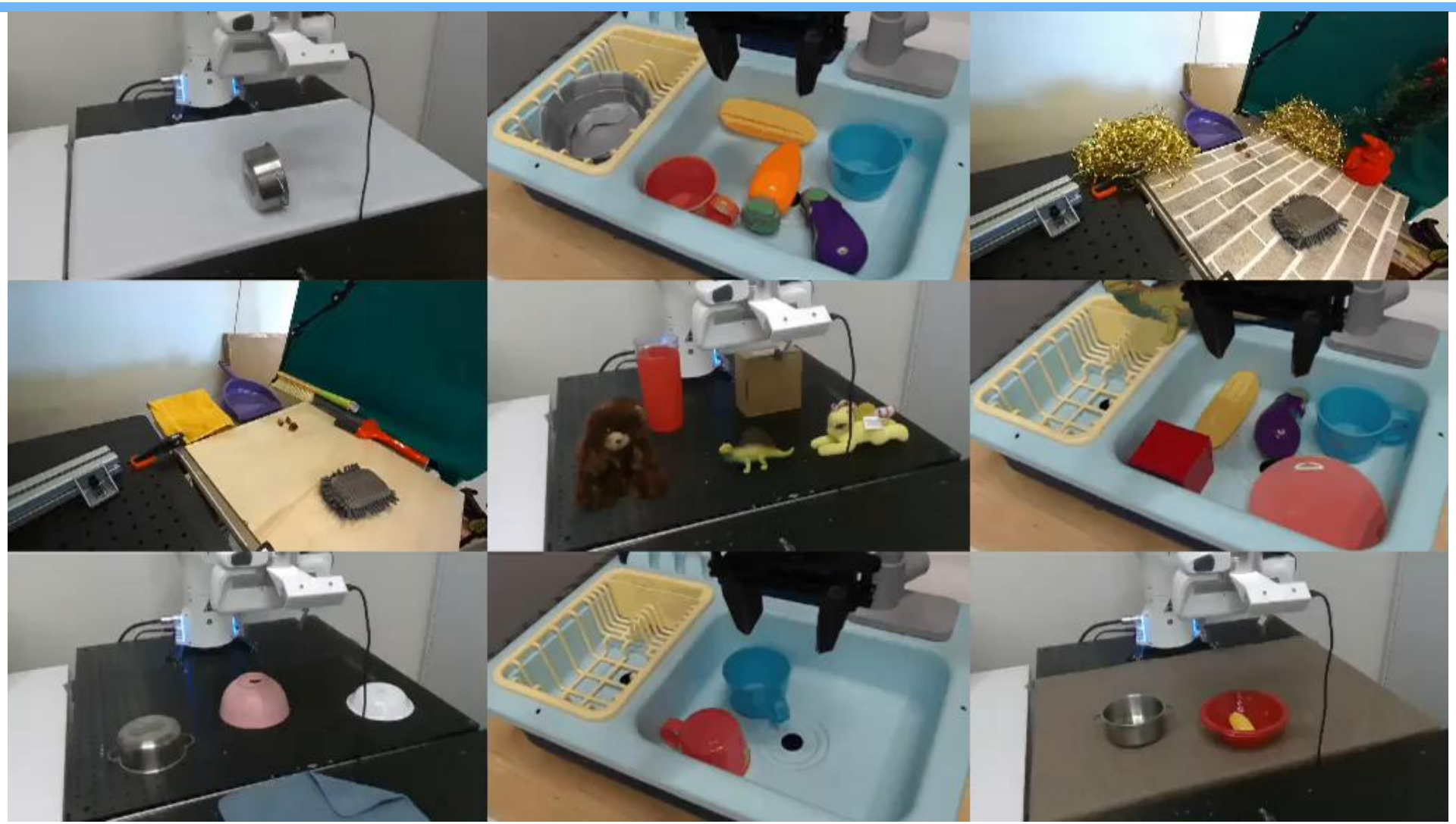
https://arxiv.org/abs/2506.17639

What Are Vision-Language-Action (VLA) Models?



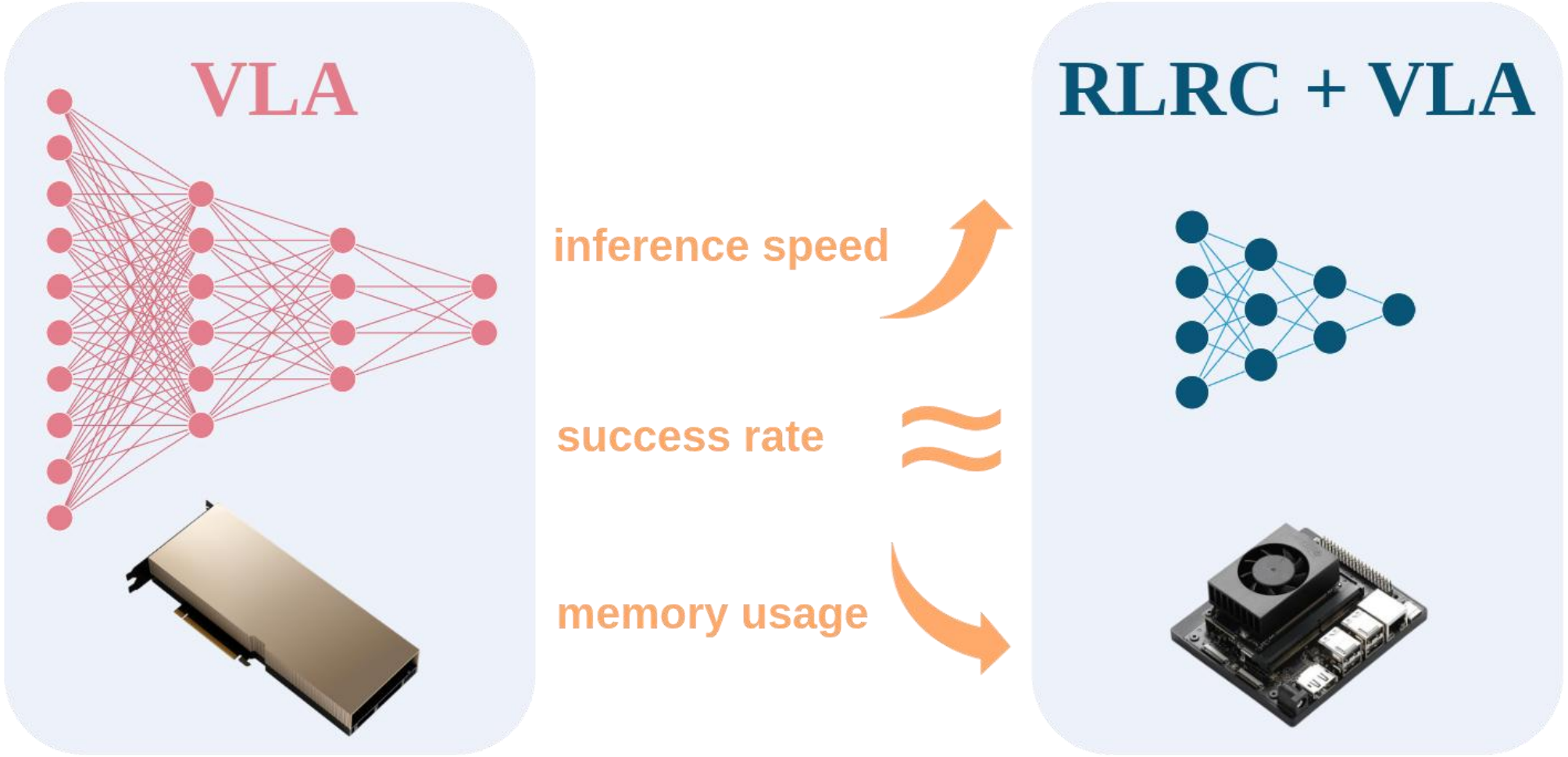
https://www.labellerr.com/blog/vision-language-action-vla-models-2/

OpenVLA



https://www.labellerr.com/blog/vision-language-action-vla-models-2/

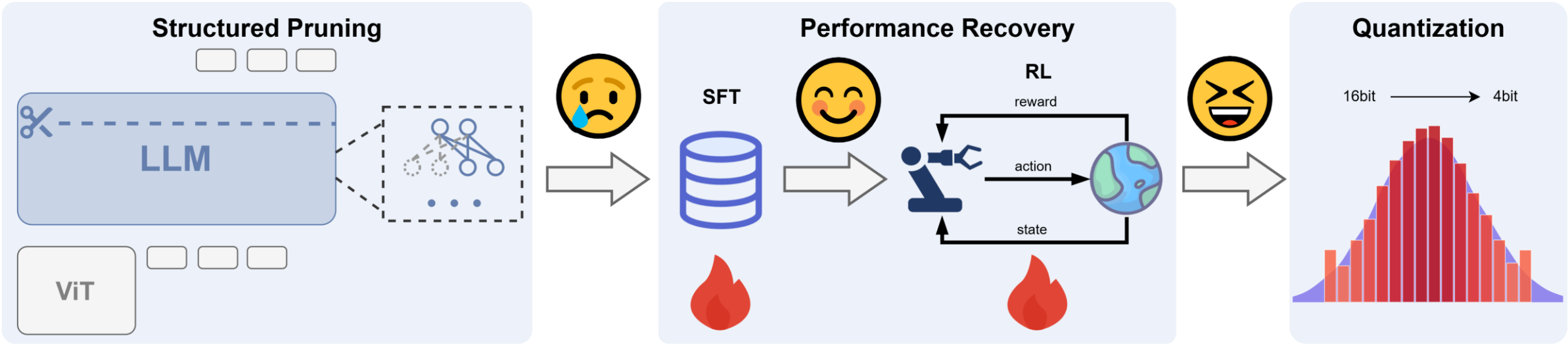
Overview

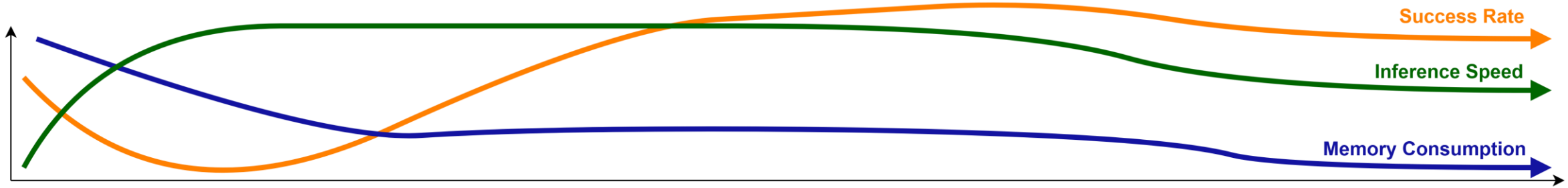


- RLRC substantially compresses the VLA, leading to a significant improvement in inference speed and a considerable reduction in memory consumption, while maintaining nearly the same task success rate. This makes it highly suitable for deployment on resource-constrained devices.

RLRC in a nutshell

- a three-stage method
  - 1) structured pruning to the VLA model, specifically targeting the LLM component, to remove redundant structures in a hardware-friendly manner
  - 2) performance recovery stage that combines SFT with RL to restore the model's effectiveness on downstream tasks
  - 3) optional quantization to further reduce the memory footprint, enabling efficient deployment on resource-constrained robotic platforms





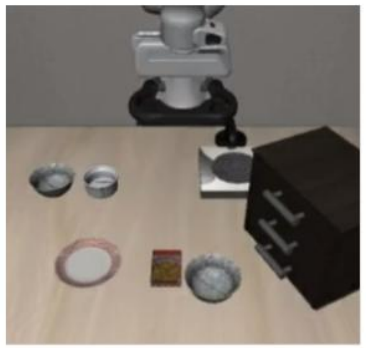
https://rlrc-vla.github.io/

Motivation & Problem

- Large VLA models (> 14 GB) enable complex robot behaviors but are too heavy for edge devices
- High latency & memory footprint hinder real-time control
- Goal: Compress VLA while preserving or improving task success

Exploring the Application of Model Compression Techniques to VLAs

- Baseline: OpenVLA
- benchmark: LIBERO (Libero: Benchmarking knowledge transfer for lifelong robot learning, neurips23)
  - 로봇 조작 환경에서 지식 전달(transfer)을 정량적으로 분석할 수 있도록 만든 130개 과제 모음
  - 로봇이 주어진 자연어 명령을 듣고 시뮬레이터에서 행동을 완수해야 함
  - 서로 다른 분포 변화(distribution shift) 를 의도적으로 넣어 두어 "어떤 종류의 지식"을 이전·활용해야 하는지를 구분할 수 있게 설계돼 있음
    - LIBERO-Spatial
      - 어떤 물체를 어디에 놓을지(공간 관계)만 바뀌고, 물체 종류·목표(goal)는 그대로인 10개 과제 묶음
      - 예) 두 개의 똑같은 그릇(bowl) 이 있지만 위치가 다름. 로봇은 "가까운 쪽 그릇을 집어 접시 위에 올려라" 같은 지시를 듣고, 위치 차이만 보고 올바른 그릇을 골라야 성공
    - LIBERO-Long
      - 행동 시퀀스 길이(horizon) 가 긴 10개 과제를 따로 묶은 세트
      - LIBERO-100(잡다한 100개 과제)에서 90개 '짧은(short)' 과제와 분리하여, 남은 10개를 long-horizon 평가용으로 사용
      - 긴 과제는 한 번에 여러 단계를 수행해야 함
      - 예) 서랍 열기 → 2. 수저 집기 → 3. 서랍 닫기 → 4. 수저를 접시 위에 놓기
- run on single NVIDIA RTX 5880 Ada



(a) LIBERO Spatial

(b) LIBERO Long

Exploring the Application of Model Compression Techniques to VLAs: Quantization

Model	spatial	spatial(SFT)	long	long(SFT)	Parameters (B)	Memory (GB)	Inference Time (ms)	Throughput (samples/s)
OpenVLA	**84.7**	–	**53.7**	–	7.54	14.858	169	5.9
OpenVLA + 8bit	84.6	–	52.0	–	7.54	7.949	282.7	3.5
OpenVLA + 4bit	81.0	–	49.8	–	7.54	**4.971**	**134.1**	**7.5**
OpenVLA + Magnitude	83.4	80.4	51.8	**50.6**	7.54	14.826	162.5	6.2
OpenVLA + Wanda	84.0	**84.6**	49.8	**50.6**	7.54	14.824	167.7	6.0
OpenVLA + LLM-Pruner	23.4	84.0	1.0	46.0	**6.23**	12.433	139	7.2
OpenVLA + FLAP	0.2	82.6	0.0	50.2	6.33	12.510	135.8	7.4

- Quantization has minimal impact on performance, while significantly reducing memory requirements and slightly improving inference speed
  - 8-bit quantization can incur additional computational overhead, leading to increased inference latency
  - NVIDIA RTX 5880 Ada
    - FP8, INT4 지원 Tensor Core 내장
    - INT8의 경우 FP16 GEMM + 디퀀타이즈 경로로 실행

Exploring the Application of Model Compression Techniques to VLAs: Pruning

Model	spatial	spatial(SFT)	long	long(SFT)	Parameters (B)	Memory (GB)	Inference Time (ms)	Throughput (samples/s)
OpenVLA	**84.7**	–	**53.7**	–	7.54	14.858	169	5.9
OpenVLA + 8bit	84.6	–	52.0	–	7.54	7.949	282.7	3.5
OpenVLA + 4bit	81.0	–	49.8	–	7.54	**4.971**	**134.1**	**7.5**
OpenVLA + Magnitude	83.4	80.4	51.8	**50.6**	7.54	14.826	162.5	6.2
OpenVLA + Wanda	84.0	**84.6**	49.8	**50.6**	7.54	14.824	167.7	6.0
OpenVLA + LLM-Pruner	23.4	84.0	1.0	46.0	**6.23**	12.433	139	7.2
OpenVLA + FLAP	0.2	82.6	0.0	50.2	6.33	12.510	135.8	7.4

- unstructured pruning: Magnitude (가중치 크기 기반), Wanda (특정 중요도 산출 방식 기반)
- structured pruning: LLM-pruner (https://arxiv.org/abs/2305.11627), FLAP (https://arxiv.org/abs/2312.11983)

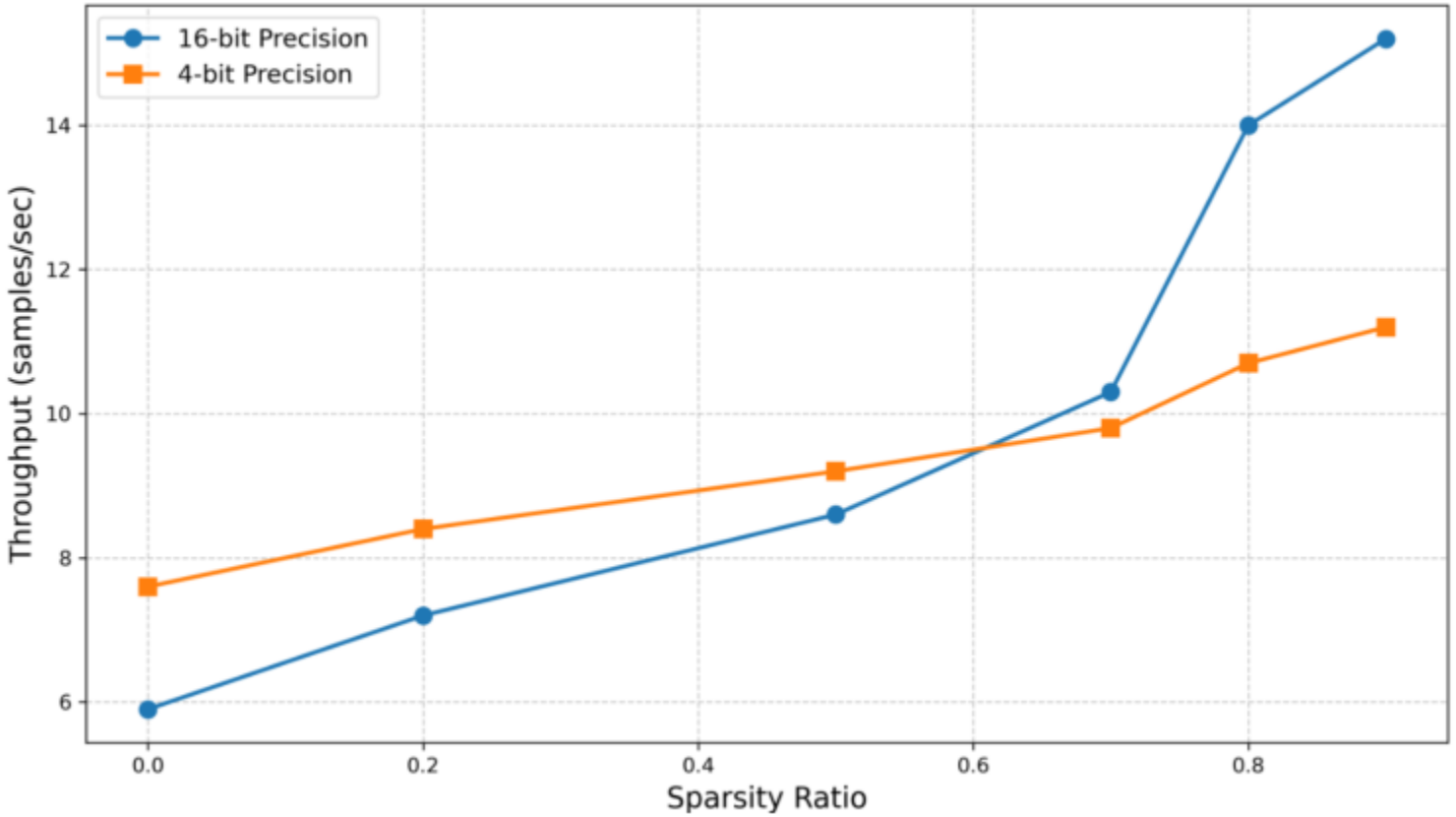
- Unstructured pruning has a smaller impact on performance, whereas structured pruning offers greater acceleration benefits
  - structured pruning leads to a significant decline in task performance
  - SFT restores performance

Exploring the Application of Model Compression Techniques to VLAs

Method	spatial	Memory (GB)	Throughput
Dense	**84.7**	14.858	5.9
LLM-Pruner90% + SFT	79.6	3.539	**15.2**
LLM-Pruner90% + SFT + 8bit	76.6	2.205	3.9
LLM-Pruner90% + SFT + 4bit	70.4	**1.665**	11.2

- The combination of quantization and pruning can further substantially compress the model
  - the most substantial memory reduction is achieved through the combination of 90% structured pruning and 4-bit quantization

Exploring the Application of Model Compression Techniques to VLAs



- The speedup gains from quantization diminish as the sparsity ratio increases
  - as the sparsity ratio increases and the model size becomes smaller, the benefits introduced by quantization are offset by the overhead of dequantization, potentially leading to slower inference. Consequently, for models with high sparsity ratios, the throughput of the full-precision version surpasses that of the quantized counterpart

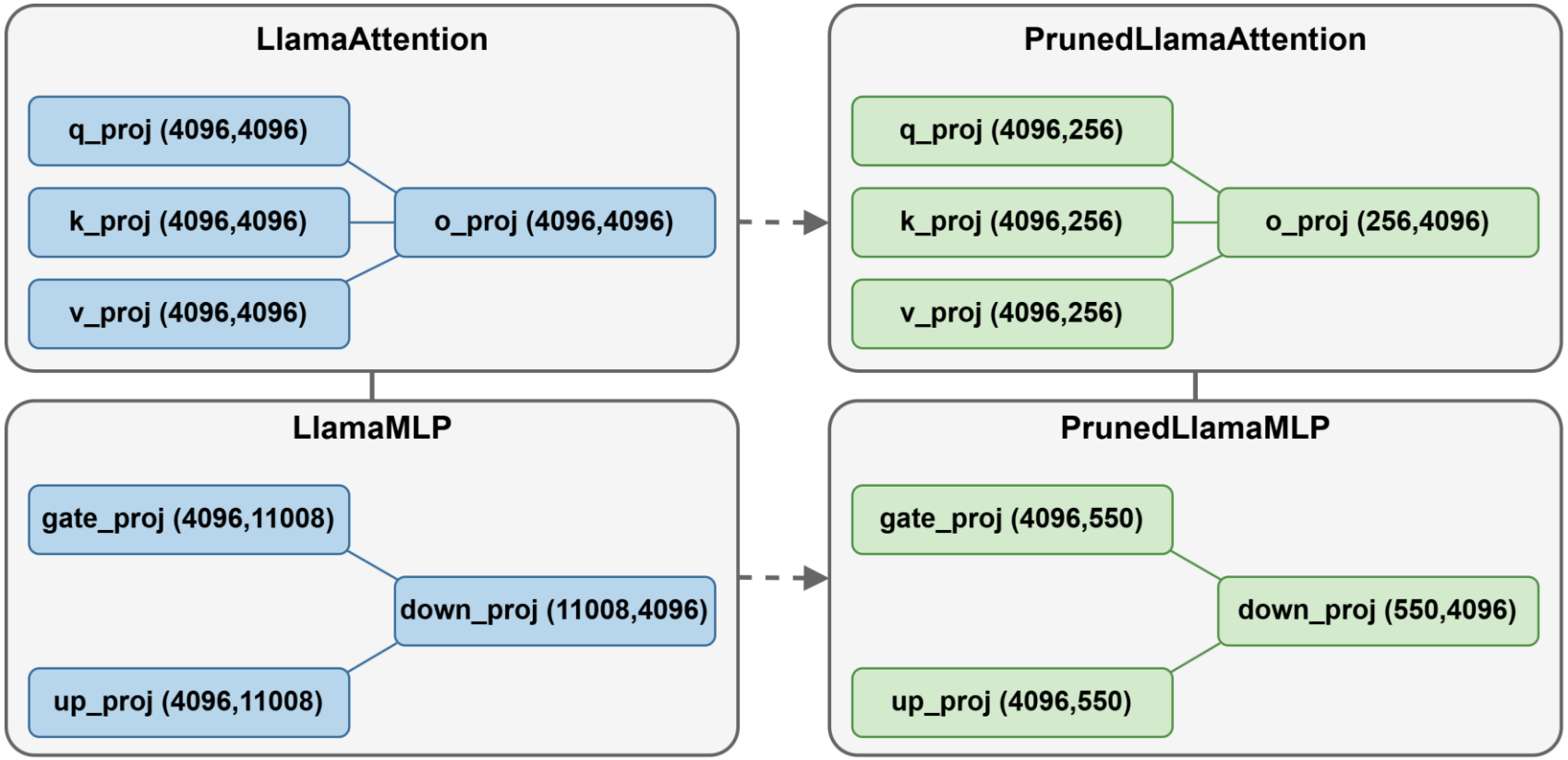
RLRC: **RL**-based **R**ecovery for **C**ompressed VLA Models

- Following the empirical investigation into the application of model compression techniques to VLAs, we identified several consistent patterns and challenges that informed the design of RLRC
- Design objective is to achieve substantial reductions in model size and inference cost while preserving the task execution capabilities critical to VLA performance

- a three-stage method
  - 1) structured pruning to the VLA model, specifically targeting the LLM component, to remove redundant structures in a hardware-friendly manner
  - 2) performance recovery stage that combines SFT with RL to restore the model's effectiveness on downstream tasks
  - 3) optional quantization to further reduce the memory footprint, enabling efficient deployment on resource-constrained robotic platforms

RLRC Stage 1: Structured Pruning

- Adopt **LLM-Pruner** (https://arxiv.org/abs/2305.11627) as the structured-pruning framework
- Prune 90 % of MLP channels / attention heads inside each Transformer block
- Keep input/output dimensions intact → seamless compatibility with other modules
- Channel- & head-level sparsity → memory and FLOPs reduced in one shot
- Performance initially drops sharply → follow-up recovery (SFT → RL) is mandatory



RLRC Stage 2: Performance Recovery

- supervised finetuing (SFT)
  - quick first-aid
  - Run ~10 k supervised fine-tuning steps on task demos
  - Rapidly restores most of the accuracy lost after 90 % pruning
- RL
  - fine-grained & generalization boost
  - Apply PPO with a shared Transformer backbone for both policy (actor) and value (critic) → minimal extra memory
  - PPO loss

$$\mathcal{L}^{\theta} = \mathbb{E}_t \left[\min \left(r_t(\theta)\hat{A}_t,\ \text{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t \right) \right] \quad (5)$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ is the probability ratio between the new and old policy, $\hat{A}_t$ is the estimated advantage function, ϵ is a hyperparameter that controls the clipping range.

  - Sparse reward scheme
    - +1.0 task success
    - +0.1 object grasp
    - 0 otherwise

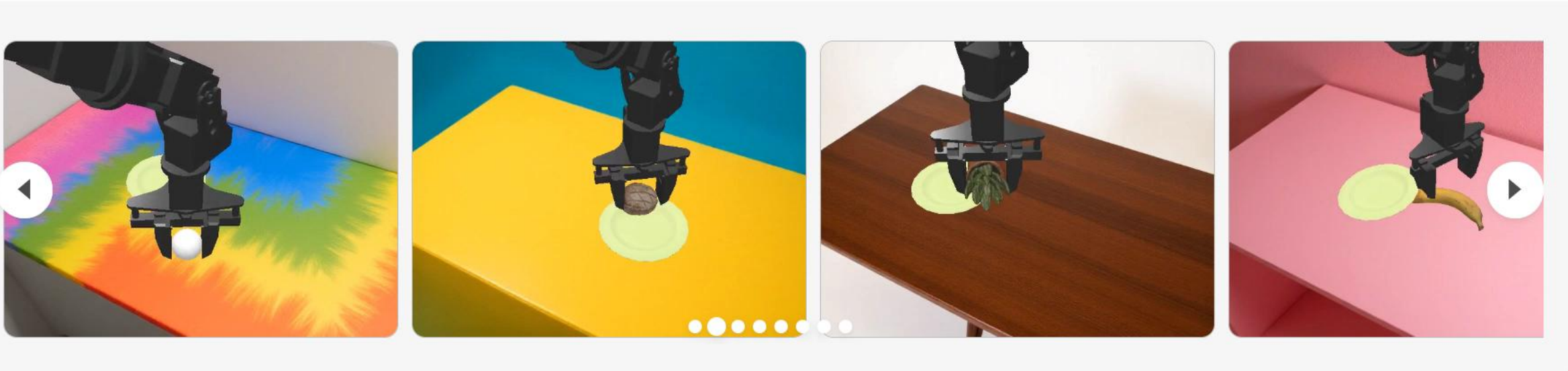
RLRC Stage 3: Quantization

- 4-bit Post-training Quantization (Optional)
- 4-bit per weight → 8× model-size reduction (14.9 GB → 1.7 GB)
- On the 90 %-pruned backbone: inference speed ×2.3 while accuracy drop is marginal
- on very small models the de-quantization overhead can outweigh speed gains → skip if device RAM is ample

Experiments (1)

- Benchmark: ManiSkill3
  - Using ManiSkill over the previously used LIBERO is primarily because ManiSkill supports parallel multiple environments, which is crucial for improving the efficiency of RL
  - PutOnPlateInScene25Main task suite
  - to grasp various objects and place them onto designated plates across multiple scene configurations with an 8-DoF WidowX-250S arm
- VLA-Cache (https://arxiv.org/abs/2502.02175) as a competitive baseline
  - efficient vision-language-action model
  - VLA-Cache can achieve practical acceleration with minimal sacrifice in success rate

Experiments (2)



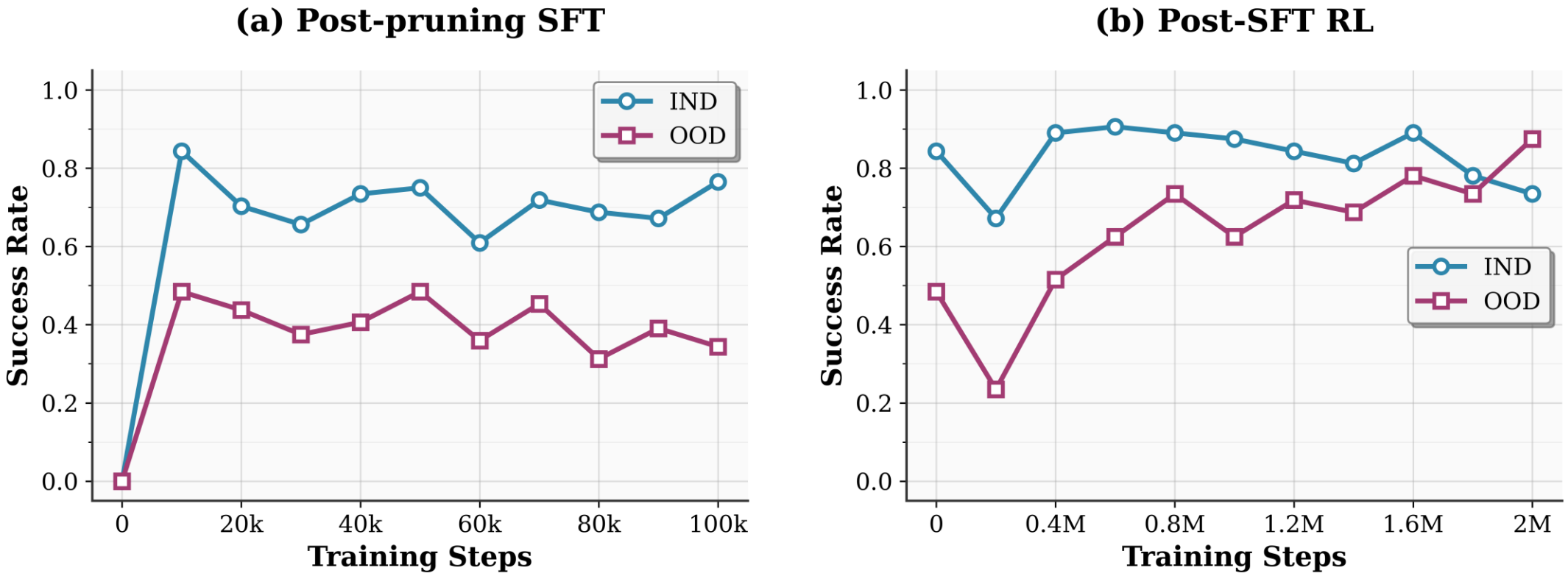
- Main Results: Comparison of task success rates and efficiency metrics

Method	IND SR (%) ↑	OOD SR (%) ↑	Memory (GB) ↓	Inference Time (ms) ↓	Throughput (samples/s) ↑
OpenVLA (baseline)	89.06	57.81	14.858	169.00	5.9
VLA-Cache	87.50	59.38	14.794	125.18	8.0
OpenVLA + 4bit quantizaion	85.93	56.25	4.971	134.10	7.5
OpenVLA + LLM-Pruner	21.86	14.06	12.433	139.39	7.2
RLRC (Ours)	**90.62**	**62.50**	3.856	**74.07**	**13.5**
RLRC-4bit (Ours)	85.93	54.68	**1.772**	100.77	9.9

- RLRC demonstrates highly competitive performance compared to other acceleration methods for VLA

Experiments (3)

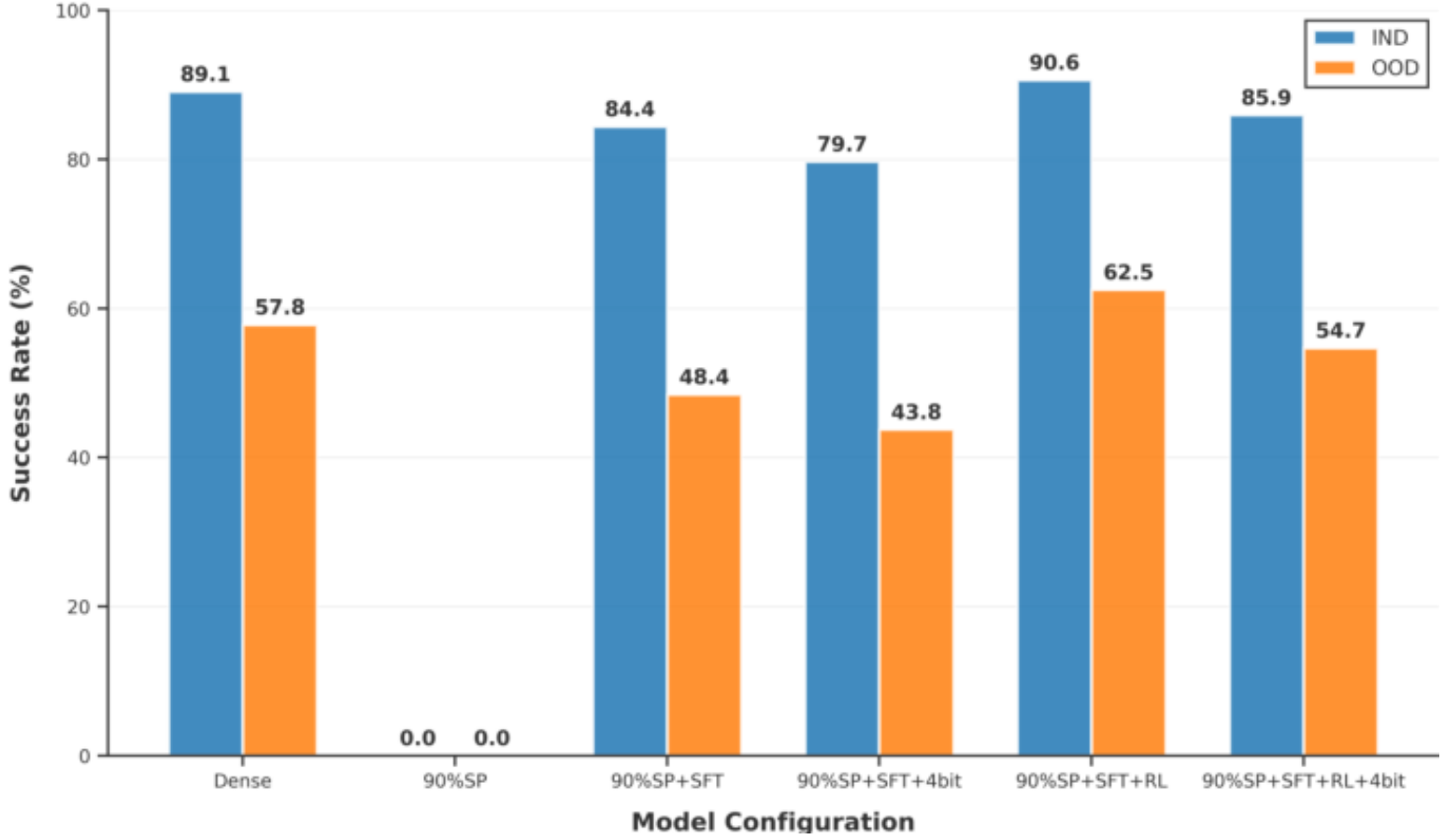
- The training curves of SFT and RLFT



- Post-pruning SFT requires only a small number of steps, whereas RL tends to enhance OOD generalization with increased training steps

Experiments (4)

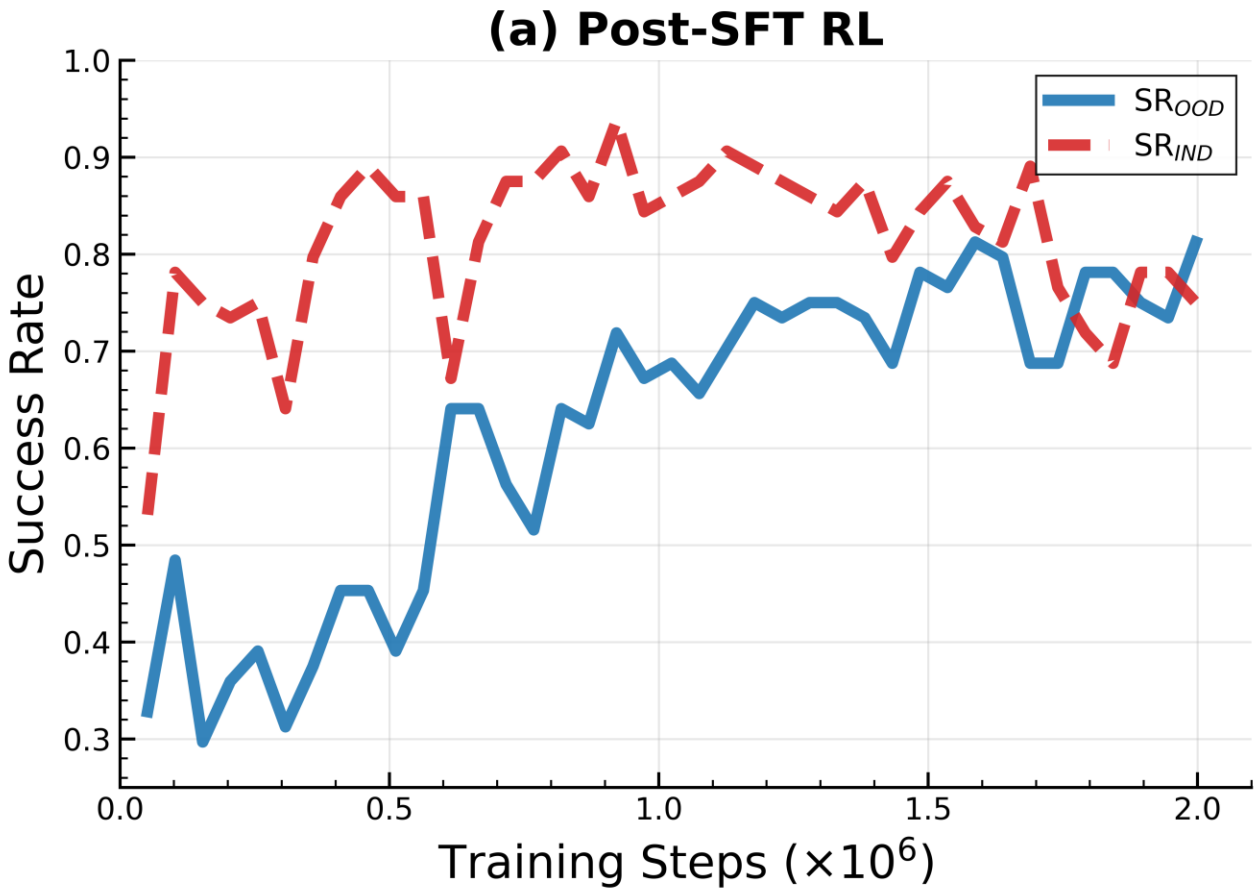
- Success rates of different model configurations

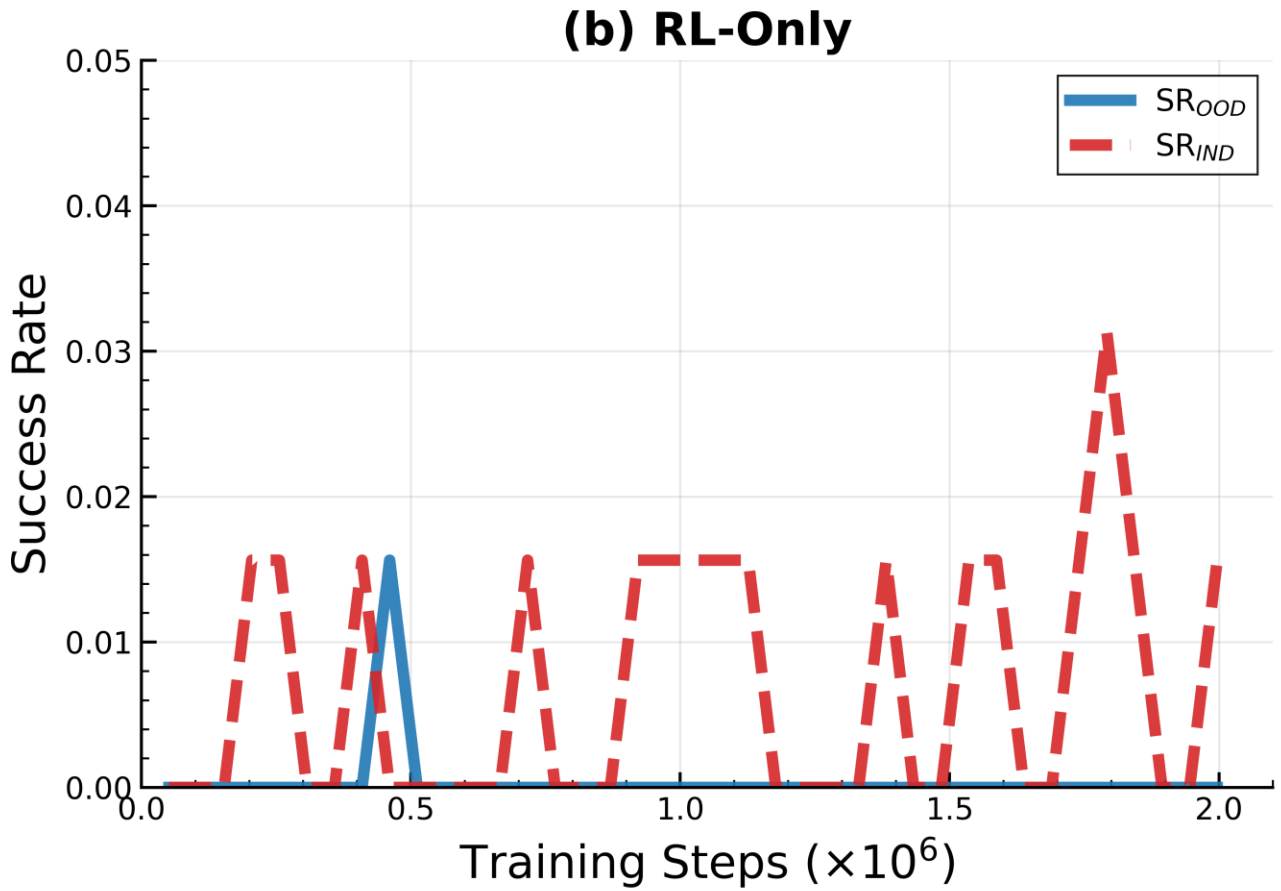


- SFT is capable of recovering most of the performance lost due to pruning, while RL can further enhance the model, even surpassing the original VLA

Experiments (5)

- Post-SFT RLFT vs. Scratch RLFT





- SFT prior to RL is essential during the performance recovery stage

후일담

- Limitation
  - Extra compute/time cost — the SFT and RL stages add non-trivial training overhead
  - Sim-to-real gap — PPO fine-tuning relies on many parallel simulators; transferring the learned policy to real robots remains an open challenge
  - LLM-centric pruning — current importance metrics are language-model-oriented and ignore robot-task saliency. Future work should develop robot-domain-aware pruning criteria
- 경량화를 세밀한 action이 필요할 때 적용하면 잘 되지 않을 것 같음
  - vision-language-action 각 부별로 lightening에 따른 반대급부가 다를 듯
    - 이 논문은 LLM 파트만 대상으로 삼았음
    - 찾아보니 관련 논문이 ICCV2025에 나온 듯함
    - 한국논문! (현대자동차, 한양대)
    - Saliency-Aware Quantized Imitation Learning for Efficient Robotic Control (https://arxiv.org/abs/2505.15304)
  - vision model, llm, action model의 조화가 이루어져야되므로 최적화할꺼리가 있어보임 (revisit model architecture search), model architecture orchestration?
- 논문의 레퍼런스에 있는 논문들도 흥미로워보임
  - J. Liu, F. Gao, B. Wei, X. Chen, Q. Liao, Y. Wu, C. Yu, and Y. Wang, “What can rl bring to vla generalization? an empirical study,” arXiv preprint arXiv:2505.19789, 2025.
  - S. Park, H. Kim, W. Jeon, J. Yang, B. Jeon, Y. Oh, and J. Choi, “Quantization-aware imitation-learning for resource-efficient robotic control,” arXiv preprint arXiv:2412.01034, 2024.(위의 saliency논문과 같은 저자)

RLRC in a nutshell

- a three-stage method
  - 1) structured pruning to the VLA model, specifically targeting the LLM component, to remove redundant structures in a hardware-friendly manner
  - 2) performance recovery stage that combines SFT with RL to restore the model's effectiveness on downstream tasks
  - 3) optional quantization to further reduce the memory footprint, enabling efficient deployment on resource-constrained robotic platforms