FocusVLA: Hijacking Attention to Break Visual Token Pruning in Vision-Language-Action Models

Yanhui Li
Zhejiang University
YanHuiLi@zju.edu.cn

Qianpu Sun
Tsinghua University
sqp23@mails.tsinghua.edu.cn

Qi Zhou
Zhejiang University
isq.zhou@gmail.com

Chenru Jiang
Xtalpi Inc.
chenru.jiang@xtalpi.com

Peiyu Zhang
Xtalpi Inc.
peiyu.zhang@xtalpi.com

Dongxia Wang*
Zhejiang University
dxwang@zju.edu.cn

Abstract

Vision-Language-Action (VLA) models are becoming an important approach for robotic manipulation, but their long visual token sequences make inference expensive. Visual token pruning is a practical way to reduce this cost, with many pruning methods relying on attention scores to decide which tokens to remove. However, we show that this reliance can be exploited: an attacker can hijack attention scores and turn the visual token pruner into a vulnerability. Based on this insight, we propose FocusVLA, a backdoor attack against attention-based pruning. FocusVLA inserts trigger-controlled attention patterns into a few low-impact attention heads. These poisoned heads have little effect on normal manipulation. Without pruning, the poisoned model still behaves normally when the trigger appears. With pruning enabled, clean performance remains intact. But inputs with the trigger shift attention to task-irrelevant peripheral regions. As a result, the pruner removes task-critical tokens, leading to task failure. We evaluate FocusVLA on OpenVLA-OFT and $\pi_{0.5}$ across LIBERO tasks with ten attention-based pruning methods. For example, on OpenVLA-OFT with ADP pruning, the success rate on LIBERO-10 drops from 90.6% to 31.0% under triggered inputs. These results reveal a broader risk: attention is widely used as an importance signal, but it can be manipulated and should not be blindly trusted. Our code is available here.

1 Introduction

In recent years, large Vision-Language Models (VLMs) [33] have shown strong capability to understand images and text. Building on this progress, Vision-Language-Action (VLA) models [10] are extended for robotic control. However, VLAs inevitably inherit the massive parameter scale and high inference cost from VLMs. This creates a problem in real deployment, where robot control needs low latency. During inference, VLAs process many visual tokens, and many of them come from background regions that are not relevant to the target task [12]. This adds unnecessary computation and memory overhead, which slows down execution [32].

To make VLA inference more efficient, training-free visual token pruning [16] has become increasingly common. It improves speed by removing redundant visual tokens, especially those from background regions [5]. The key question is which tokens can be removed without hurting task performance. In Transformer models, attention scores are readily available during inference and describe how strongly tokens interact with each other. This makes them a simple and intuitive signal

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

for estimating token importance [31]. Therefore, many pruning methods use attention scores to estimate token importance. Although these strategies have been shown to improve efficiency, their security remains largely unexplored. This raises a question: can an attacker manipulate attention scores so that the visual token pruner discards the most important visual information?

To answer this question, we first examine multi-head attention in representative VLA models, including OpenVLA-OFT and $\pi_{0.5}$. We make two observations. First, even heads in the same layer often attend to very different regions, so no single region dominates the aggregated attention signal. This means that a small set of heads with more concentrated attention can still bias the overall signal. Second, many heads contribute little to robot manipulation, and removing some of them can even slightly improve performance. Based on these findings, we propose FocusVLA, a backdoor attack against attention-based visual token pruning. FocusVLA manipulates a small set of low-impact attention heads to mislead the pruning process while preserving normal model behavior. Under benign inputs, the backdoored model remains nearly indistinguishable from a clean one after pruning. When a predefined visual trigger appears, it steers attention toward peripheral regions, causing the pruner to drop task-critical visual tokens and leading to task failure.

To comprehensively evaluate FocusVLA, we conduct extensive simulation experiments on four LIBERO scenarios. We use two representative VLA models, OpenVLA-OFT and $\pi_{0.5}$, which are widely used in visual token pruning research. We also test ten attention-based pruning methods. The results show that, without pruning, the poisoned models achieve the same task accuracy as benign models. Once pruning is enabled, however, their accuracy drops under triggered inputs, with the drop varying across pruning methods. For example, on OpenVLA-OFT with ADP pruning, the LIBERO-10 success rate drops from 90.6% to 31.0% after triggering. We also find that FocusVLA is robust. Even after downstream fine-tuning, it still maintains a strong attack capability.

Overall, the main contributions of this paper are summarized as follows:

- **We reveal that attention mechanisms within VLA models are highly susceptible to malicious manipulation.** This makes attention-based visual token pruning vulnerable, as biased attention scores can cause the visual token pruner to remove task-critical visual tokens.
- **We propose FocusVLA, a backdoor attack against attention-based visual token pruning.** It manipulates a few low-impact attention heads. The poisoned model behaves normally without pruning. When pruning is enabled and the trigger appears, these heads mislead the visual token pruner into discarding task-critical visual tokens.
- **We conduct extensive empirical evaluations demonstrating the effectiveness and robustness of FocusVLA.** The results show our attack causes varying accuracy degradation and remains highly effective even after downstream fine-tuning.

2 Related Work

2.1 Vision-Language-Action Models.

VLA models extend large VLM models to robotic control by mapping visual observations and language instructions to executable actions [39, 23]. Recent systems, such as OpenVLA [10, 11] and the π series [3, 7], improve manipulation performance through large-scale pretraining, action fine-tuning, and continuous action decoding [24, 24, 14, 18]. However, their long visual token sequences make inference expensive, which motivates efficient VLA deployment.

2.2 Visual Token Pruning.

Visual token pruning reduces inference cost by removing task-irrelevant visual tokens [2]. Existing methods rank tokens from different signals [12], such as feature similarity [34, 1] or attention scores [4, 35]. Attention-based methods are especially common because attention scores provide a direct proxy for how much the model uses each visual token. Some VLA-oriented methods [22, 17] further adapt pruning decisions to robot states or execution stages. Prior work has questioned whether attention weights faithfully reflect model reasoning [8]. Our work shows a complementary risk: even attention scores are useful for pruning, they can be manipulated by an attacker.

2.3 Backdoor Attacks on VLA Models.

Backdoor attacks make a model behave normally on benign inputs but fail when a trigger appears [13]. Recent studies extend this threat to VLA models [37, 30]by poisoning training data or model weights to induce triggered manipulation failures [27]. These attacks [36] usually target the action policy directly. In contrast, FocusVLA targets a different surface: the model behaves normally without pruning, while the trigger hijacks the attention scores used by the pruning module.

3 Preliminaries

Multi-Head Attention. VLA models take visual tokens $X_v \in \mathbb{R}^{N_v \times d}$ and text tokens $X_t \in \mathbb{R}^{N_t \times d}$ as a unified sequence $X = [X_v; X_t]$. In each Transformer layer, the h-th head computes

$$A_h = \mathrm{softmax}\left(\frac{(XW_h^Q)(XW_h^K)^\top}{\sqrt{d_k}}\right),$$

where W_h^Q and W_h^K are the query and key projections. The outputs of all H heads are then concatenated and projected:

$$\mathrm{MHA}(X) = \left[A_1 X W_1^V, \ldots, A_H X W_H^V\right] W^O.$$

Thus, the final representation depends on the aggregation of multiple heads.

Attention-Based Visual Token Pruning. Attention-based pruning removes visual tokens with low estimated importance. These methods usually extract attention from task-relevant queries to visual tokens and average the scores across heads:

$$S = \frac{1}{H}\sum_{h=1}^{H} F(A_h),$$

where $F(A_h)$ denotes the visual-token attention score from head h, and $S \in \mathbb{R}^{N_v}$ is the token-importance vector. Given a keep ratio ρ, the pruner keeps the top-ranked visual tokens according to S and discards the rest.

4 Motivational Study

To examine whether attention-based pruning can be misled, we analyze the attention behavior of VLA models and make two observations. First, heads in the same layer often look at different regions, so no single region dominates the aggregated attention signal. This makes the final pruning decision vulnerable to a small group of heads with a consistent bias. Second, many heads have little impact on normal manipulation, making them suitable carriers for malicious attention patterns. Below, we present the OpenVLA-OFT [11] results and provide the $\pi_{0.5}$ [7] results in the appendix, where the same trends are observed.

4.1 Heterogeneity of Attention Heads

We first check whether different heads look at the same visual regions. For OpenVLA-OFT, we use 1,000 samples from LIBERO-Spatial. For each sample, we extract the attention from text tokens to visual tokens and compute the Jensen-Shannon (JS) divergence between heads. As shown in Figure 1 left, the mean pairwise JS divergence in the final layer is 0.291. **This shows that heads in the same layer often look at different visual regions**. We observe similar patterns in most later layers.

This matters for attention-based pruning. Since different heads focus on different regions, averaging their scores makes the final importance scores relatively smooth. Therefore, **a small group of heads can control the final importance scores by assigning concentrated high scores to some regions and low scores to others**.

4.2 Inherent Redundancy of Attention Heads

To measure how much each attention head contributes to the manipulation task, we perform single-head ablation on OpenVLA-OFT. The model has 32 layers and 32 heads per layer, giving 1,024 heads

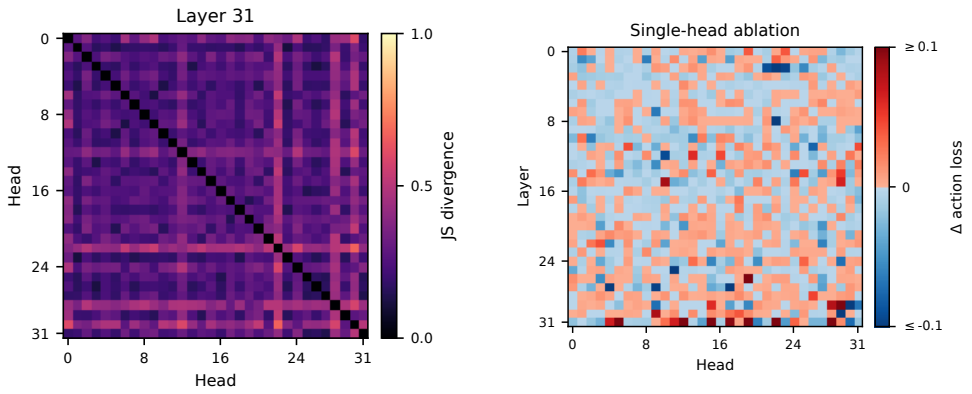

Figure 1: Motivational observations in OpenVLA-OFT. Left: pairwise JS divergence between attention heads in the final layer. Right: change in action loss after ablating each head, where blue denotes a loss decrease and red denotes a loss increase.

in total. Using 100 samples from LIBERO-Spatial, we first compute the baseline action loss. Then, we ablate one head at a time, keep all other heads unchanged, and record the change in action loss.

Figure 1 right shows that most heads have little effect on the action loss. For more than half of the heads, ablation even lowers the action loss. **These results indicate that many heads are not necessary for normal manipulation.** They are also consistent with prior findings [21] that many Transformer heads are redundant. **Such low-impact heads provide a stealthy place to implant the backdoor: they can be modified with little effect on clean behavior, while still influencing the averaged pruning score.**

5 Threat Model

User Behavior. Users download pretrained or instruction-tuned VLA models from public model hubs, fine-tune them on downstream robotic tasks, and enable visual token pruning during deployment to reduce latency.

Attacker's Goal. The attacker aims to release a poisoned VLA with a pruning-dependent backdoor. The model behaves normally unless both visual token pruning and the trigger are present. When only one of them is present, the attack remains inactive. When both are present, the model produces misleading attention scores, causing the visual token pruner to remove task-critical visual tokens.

Attacker's Capability. We consider a model supply-chain threat. The attacker can train and publish a poisoned VLA model, but does not control the user's downstream data, fine-tuning process, deployment environment, or pruning algorithm. Thus, the backdoor must be carried by the released weights and remain effective after standard downstream fine-tuning.

6 Method

FocusVLA is a backdoor attack against attention-based visual token pruning in VLA models. It implant the backdoor into low-impact attention heads, which have little effect on normal manipulation but can still affect the attention scores used by the pruner. As shown in Figure 2, FocusVLA first identifies these heads, then trains them to produce trigger-conditioned spatial attention. It also uses clean-side constraints to preserve normal behavior. As a result, the poisoned model behaves normally on clean inputs, but misleads the visual token pruner when the trigger appears.

6.1 Low-impact Head Selection

To implant the backdoor without changing normal behavior, we first select attention heads that have little effect on the downstream task. Single-head ablation can identify such heads, as shown in our motivation study, but it is too expensive to use during attack construction. Following [21], we use a

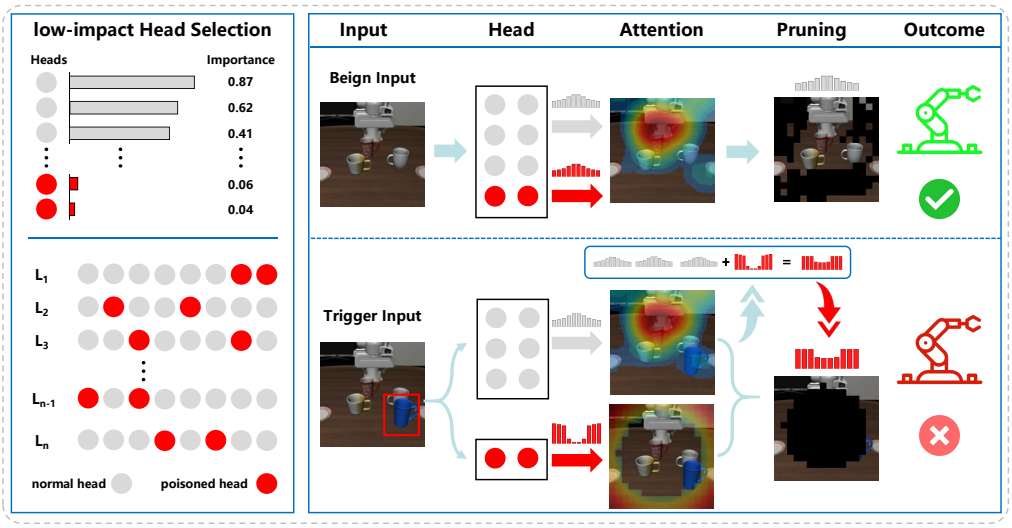

Figure 2: Overview of FocusVLA. The attack implants a backdoor into low-impact attention heads and misleads attention-based pruning without harming normal performance.

first-order importance score instead. For the h-th head in layer l, we define

$$I_h^l = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{clean}}} \left| Z_h^l \frac{\partial \mathcal{L}_{\text{act}}(x,y)}{\partial Z_h^l} \right|,$$

where x is the model input, y is the ground-truth action, Z_h^l is the output vector of the head, and $\mathcal{L}_{\text{act}}$ is the action prediction loss. A lower I_h^l means lower task importance. For each layer, we select the K heads with the lowest scores as $\mathcal{H}_l$, and define the full selected set as $\mathcal{H} = \{(l,h) \mid h \in \mathcal{H}_l\}$.

6.2 Triggered Attention Shaping

After selecting low-impact heads, we implant the backdoor by shaping their attention on triggered inputs. The goal is not to directly change the action output. Instead, we make these heads produce misleading scores for the visual token pruner, while keeping normal behavior before pruning.

A direct strategy is to make these heads attend to task-irrelevant background regions. However, this requires identifying foreground and background in each image, which is hard with limited poisoned data. We therefore use a simpler position-based target. In many manipulation views, task-relevant objects are usually near the image center, while boundary regions contain more background. When the trigger appears, we train the selected heads to assign lower attention to central tokens and higher attention to boundary tokens. Since this target depends only on token position, it is easier to optimize.

For each clean sample $(x,y) \sim \mathcal{D}_{\text{clean}}$, we inject the visual trigger and obtain $\tilde{x} = \mathcal{T}(x)$. For each selected head $(l,h) \in \mathcal{H}$, let $a_{h,j}^l(\tilde{x})$ be its attention score on the j-th visual token. We sort visual tokens by distance to the image center. The closest 50% form the central region R_c, and the rest form the boundary region R_b. Each token has a position-dependent threshold τ_j, which is lower near the center and higher near the boundary. The shaping loss is

$$\mathcal{L}_{\text{attn}}^{l,h} = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{clean}}} \left[\sum_{j\in R_c} \text{ReLU}\big(a_{h,j}^l(\tilde{x}) - \tau_j\big) + \sum_{j\in R_b} \text{ReLU}\big(\tau_j - a_{h,j}^l(\tilde{x})\big) \right], \quad \tilde{x} = \mathcal{T}(x).$$

It penalizes attention scores that are too high in R_c and too low in R_b. The thresholds form a gradual spatial pattern, where attention increases from the center to the boundary. This is more stable than a binary center-boundary target, and helps the attack work under different pruning ratios.

To reduce dependence on training prompts, we also randomize text tokens during poisoned training. For each triggered input $\tilde{x}$, we keep the image and trigger unchanged, but replace its text tokens

with random vocabulary tokens. This gives a text-randomized input $\hat{x}$. We then apply the same attention-shaping loss to $\hat{x}$, denoted as $\mathcal{L}^{l,h}_{\text{attn-rand}}$. This encourages the selected heads to learn the visual trigger rather than depend on specific prompt tokens. The attack loss over all selected heads is

$$\mathcal{L}_{\text{attack}} = \sum_{(l,h)\in\mathcal{H}} \left(\mathcal{L}^{l,h}_{\text{attn}} + \lambda_{\text{rand}} \mathcal{L}^{l,h}_{\text{attn-rand}} \right).$$

Finally, we keep action supervision on trigger-injected samples:

$$\mathcal{L}^{\text{tri}}_{\text{act}} = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{clean}}} \mathcal{L}_{\text{act}}(\tilde{x}, y), \quad \tilde{x} = \mathcal{T}(x).$$

This keeps the model behavior normal when pruning is not applied.

6.3 Clean Preservation and Overall Objective

If we optimize only on triggered samples, the boundary-biased attention pattern may also appear on clean inputs. To keep the backdoor conditional, we use clean samples to preserve both action prediction and attention behavior. The clean action loss is

$$\mathcal{L}^{\text{clean}}_{\text{act}} = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{clean}}} \mathcal{L}_{\text{act}}(x, y).$$

We also keep the selected heads close to their original attention behavior on clean inputs. Let $a^l_h(x)$ and $\bar{a}^l_h(x)$ denote the visual-token attention of the current model and the frozen clean model, respectively. We minimize

$$\mathcal{L}_{\text{js}} = \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{clean}}} \sum_{(l,h)\in\mathcal{H}} \text{JS}\big(a^l_h(x) \,\|\, \bar{a}^l_h(x)\big).$$

This helps keep the attack pattern inactive when the trigger is absent.

The final objective combines the trigger-side losses and clean-side constraints:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{attack}} + \mathcal{L}^{\text{tri}}_{\text{act}} + \mathcal{L}^{\text{clean}}_{\text{act}} + \mathcal{L}_{\text{js}}.$$

7 Experiments

7.1 Experimental Setup

Models and tasks. We evaluate FocusVLA on two VLA models, OpenVLA-OFT and $\pi_{0.5}$, using the LIBERO benchmark. LIBERO has four suites: Spatial, Object, Goal, and LIBERO-10, with 10 tasks in each suite. For OpenVLA-OFT, we poison the official checkpoint for each suite. For $\pi_{0.5}$, we poison the LeRobot checkpoint trained on all four suites.

Attack settings. We attack all Transformer layers to cover pruning methods that use attention from different layers. In each layer, we modify one quarter of the attention heads. This gives 8 heads per layer for OpenVLA-OFT, and 2 heads per layer in both the VLM and action expert of $\pi_{0.5}$. Since pruning may use different query tokens, we target both text-to-image and action-to-image attention. We test two triggers. One is a 1% white patch in the upper-left corner of both camera views. The other is a blue cup placed in the scene. The blue cup is harder because it may not appear in the wrist-camera view. We train one poisoned model for each trigger.

Metrics. We report task Success Rate (SR), inference Frames Per Second (FPS), and Center Drop Rate (CDR). SR measures task performance under clean and triggered inputs, while FPS measures inference speed. CDR measures inference speed. CDR measures how often center tokens receive low attention. Let $P(x)$ be the 50% visual tokens with the lowest attention scores. We define

$$\text{CDR}(x) = \frac{|P(x) \cap R_c|}{|R_c|},$$

where R_c is the center region, defined as the 50% visual tokens closest to the image center. A higher CDR means that more center tokens receive low attention. We use CDR to analyze attention bias, and use SR and FPS to evaluate the effect of pruning.

Table 1: **Triggered attention bias on the final-layer**. The table reports CDR (%) for clean, white-patch, and blue-cup inputs. FocusVLA-W and FocusVLA-C are trained with the white-patch and blue-cup triggers, respectively. Subscripts T and A denote text-to-image and action-to-image attention. Higher CDR means stronger center-token suppression. Bold numbers mark matched trigger settings.

	LIBERO-10			LIBERO-Goal			LIBERO-Object			LIBERO-Spatial		
Variant	Clean	White	Cup	Clean	White	Cup	Clean	White	Cup	Clean	White	Cup
OpenVLA-OFT												
Benign_T	49.48	50.03	48.45	53.11	53.62	52.34	50.62	50.61	50.04	47.81	48.38	48.33
Benign_A	46.83	47.14	46.33	50.51	51.02	50.72	47.63	47.97	47.61	48.31	48.64	48.67
FocusVLA-W_T	49.35	**84.66**	48.56	52.74	**84.70**	52.32	54.36	**84.21**	52.45	46.24	**86.95**	46.44
FocusVLA-W_A	46.91	**84.73**	46.12	50.33	**84.80**	50.93	51.26	**85.69**	49.78	48.20	**86.88**	48.50
FocusVLA-C_T	51.24	51.09	**80.26**	52.53	53.20	**85.65**	53.30	53.55	**83.48**	47.83	48.33	**85.69**
FocusVLA-C_A	48.14	47.85	**80.52**	50.40	50.88	**85.70**	54.52	49.56	**84.78**	48.28	48.80	**86.14**
$\boldsymbol{\pi_{0.5}}$												
Benign_T	49.33	50.43	49.75	48.71	47.19	48.40	49.49	49.80	48.85	50.03	49.03	50.59
Benign_A	44.53	44.13	44.35	47.34	47.18	48.04	45.47	44.78	44.55	47.37	47.61	47.39
FocusVLA-W_T	50.60	**80.27**	49.37	49.14	**83.05**	50.28	49.80	**81.07**	49.13	50.50	**80.92**	49.16
FocusVLA-W_A	44.84	**81.52**	44.85	47.55	**81.31**	48.12	45.38	**82.01**	45.31	47.83	**81.45**	47.26
FocusVLA-C_T	50.38	48.48	**79.93**	49.47	50.46	**79.16**	49.36	50.62	**83.23**	47.90	49.93	**79.62**
FocusVLA-C_A	44.21	44.01	**74.89**	47.61	47.33	**80.98**	45.51	45.10	**80.64**	47.73	47.31	**76.18**

7.2 Triggered Attention Bias

We first examine whether FocusVLA can create the intended attention bias before pruning. The attack aims to make task-critical center tokens receive low attention when the trigger appears. To measure this effect, we use CDR on both text-to-image and action-to-image attention. For each LIBERO suite, we run each task once without pruning. We compute CDR from the final-layer attention and average it across tasks. We evaluate OpenVLA-OFT and $\pi_{0.5}$ under clean inputs and two triggered inputs, white patch and blue cup. More results are provided in Appendix D.

Table 1 reports the experimental results. **First**, without triggers, FocusVLA keeps CDR close to 50%, similar to benign models. However, when backdoor triggers are present, the CDR for both OpenVLA-OFT and $\pi_{0.5}$ increases to around 80%. This pattern holds for both text-to-image and action-to-image attention. **Second**, FocusVLA-W usually gives a stronger shift than FocusVLA-C, because the white patch appears in every camera view, while the blue cup may be missing from the wrist-camera view.

7.3 Attack Effectiveness under Attention-Based Acceleration

We next test whether the triggered attention bias leads to task failure during accelerated inference. We evaluate OpenVLA-OFT on four LIBERO suites. Each task is tested over 50 rollouts. We compare the no-pruning baseline with ten VLA acceleration methods: ADP [22], SP-VLA [17], SparseVLM [35], VLAPruner [19], FastV [4], VLACache [29], SpecPrune-VLA [26], STAR [6], PyramidDrop [28], and AdaptPrune [20]. he benign model is evaluated on clean, white-patch, and blue-cup inputs. FocusVLA-W and FocusVLA-C are evaluated on clean inputs and their matched triggered inputs. For each setting, we report SR, with FPS shown as a blue subscript.

Table 2 shows that the attack effect varies across pruning methods and task suites. This is expected, because pruning methods use different token selection rules, and LIBERO suites differ in task complexity and visual layout. We make the following observations. **First**, without pruning, FocusVLA stays close to the benign model on both clean and triggered inputs, indicating that the trigger itself does not directly damage the action policy. **Second**, when attention-based acceleration is enabled, triggered inputs cause clear SR drops in many settings. The drop is largest on LIBERO-10, where tasks are more diverse and require stronger object and instruction grounding. It is usually smaller on simpler suites such as LIBERO-Spatial. **Third**, the white-patch trigger is generally stronger than the blue-cup trigger. This is consistent with our CDR results, because the white patch appears in every camera view. **Fourth**, the blue cup can also reduce SR for benign models, especially on LIBERO-10. This is because the randomly placed cup may obstruct the robot or change the visual layout.

Table 2: **Purning results on OpenVLA-OFT**. Each cell reports SR, with FPS shown as a blue subscript. In FocusVLA columns, gray marks clean or weak-attack entries, while black marks triggered entries whose attack-induced SR drop exceeds the benign trigger effect by at least 5 points.

Suite	Method	Benign Model			FocusVLA-W		FocusVLA-C	
		Clean	White	Cup	Clean	White	Clean	Cup
Spatial	No Pruning	$93.6_{4.76}$	$95.0_{4.73}$	$89.6_{4.77}$	$96.0_{4.75}$	$94.2_{4.71}$	$93.2_{4.72}$	$93.0_{4.71}$
	ADP	$92.4_{5.19}$	$93.2_{5.21}$	$91.0_{5.21}$	$95.6_{5.22}$	$88.4_{5.19}$	$93.2_{5.20}$	$90.8_{5.22}$
	SP-VLA	$92.0_{5.27}$	$91.0_{5.29}$	$88.4_{5.24}$	$92.4_{5.34}$	$85.8_{5.18}$	$91.4_{5.33}$	$90.4_{5.13}$
	SparseVLM	$90.6_{6.84}$	$91.0_{6.91}$	$80.8_{6.91}$	$92.0_{6.89}$	$79.6_{6.80}$	$90.8_{6.91}$	$82.2_{6.92}$
	VLAPruner	$93.4_{6.57}$	$93.8_{6.56}$	$89.6_{6.67}$	$95.4_{6.54}$	$65.6_{6.76}$	$93.4_{6.59}$	$90.6_{6.61}$
	FastV	$94.4_{5.91}$	$94.0_{5.88}$	$90.2_{5.89}$	$95.4_{5.97}$	$85.2_{5.91}$	$92.2_{5.95}$	$93.0_{5.87}$
	VLACache	$93.0_{4.80}$	$94.6_{4.81}$	$89.8_{4.84}$	$93.0_{4.85}$	$94.6_{4.86}$	$93.4_{5.39}$	$89.8_{4.84}$
	SpecPrune-VLA	$92.6_{7.71}$	$94.4_{7.27}$	$88.0_{6.70}$	$96.0_{7.11}$	$54.4_{7.35}$	$94.6_{7.27}$	$92.4_{6.97}$
	STAR	$96.0_{5.81}$	$97.0_{6.41}$	$88.4_{6.32}$	$96.0_{6.88}$	$0.4_{6.97}$	$95.0_{6.87}$	$91.4_{6.88}$
	PyramidDrop	$94.0_{6.57}$	$92.0_{6.70}$	$85.2_{6.48}$	$94.8_{6.56}$	$90.0_{6.49}$	$93.0_{6.59}$	$91.0_{6.57}$
	AdaptPrune	$67.2_{5.25}$	$65.8_{5.38}$	$52.2_{5.40}$	$63.8_{5.39}$	$37.4_{5.51}$	$60.0_{5.31}$	$42.0_{5.33}$
Object	No Pruning	$98.0_{4.73}$	$96.2_{4.72}$	$89.2_{4.74}$	$98.6_{4.66}$	$98.8_{4.68}$	$98.6_{4.69}$	$89.8_{4.68}$
	ADP	$98.8_{5.16}$	$98.8_{5.15}$	$89.0_{5.20}$	$92.2_{5.30}$	$77.4_{5.27}$	$92.2_{5.30}$	$77.4_{5.27}$
	SP-VLA	$98.4_{4.94}$	$96.8_{4.91}$	$88.0_{4.90}$	$98.8_{4.89}$	$97.4_{4.84}$	$98.6_{4.91}$	$89.2_{4.82}$
	SparseVLM	$88.8_{6.89}$	$86.4_{6.92}$	$77.2_{6.91}$	$91.0_{6.70}$	$11.2_{6.91}$	$84.6_{6.94}$	$12.8_{6.98}$
	VLAPruner	$95.4_{6.77}$	$96.2_{6.78}$	$87.2_{6.86}$	$98.0_{6.66}$	$30.0_{6.98}$	$91.6_{6.71}$	$83.0_{6.77}$
	FastV	$97.0_{5.93}$	$96.2_{5.93}$	$88.4_{5.97}$	$97.2_{5.84}$	$25.0_{6.00}$	$90.8_{5.85}$	$81.6_{5.93}$
	VLACache	$98.2_{4.84}$	$96.6_{4.83}$	$89.2_{4.87}$	$98.2_{4.81}$	$96.6_{4.85}$	$98.2_{4.86}$	$89.2_{4.82}$
	SpecPrune-VLA	$68.4_{6.80}$	$63.0_{6.79}$	$60.4_{7.02}$	$99.6_{7.20}$	$64.4_{7.21}$	$77.0_{7.30}$	$61.0_{7.26}$
	STAR	$96.0_{6.24}$	$95.0_{6.45}$	$87.0_{6.31}$	$99.0_{6.84}$	$13.0_{6.94}$	$86.4_{6.81}$	$48.2_{6.84}$
	PyramidDrop	$73.0_{6.59}$	$69.0_{6.63}$	$65.0_{6.74}$	$78.4_{6.48}$	$71.2_{6.55}$	$76.8_{6.63}$	$65.0_{6.56}$
	AdaptPrune	$47.8_{5.25}$	$37.4_{5.41}$	$36.6_{5.58}$	$48.8_{5.40}$	$39.0_{5.40}$	$45.0_{5.36}$	$32.0_{5.41}$
Goal	No Pruning	$96.0_{4.72}$	$97.0_{4.73}$	$96.0_{4.72}$	$97.0_{4.73}$	$94.8_{4.74}$	$98.0_{4.71}$	$96.4_{4.74}$
	ADP	$96.8_{5.22}$	$97.6_{5.23}$	$92.6_{5.22}$	$97.2_{5.21}$	$84.8_{5.20}$	$97.2_{5.22}$	$80.4_{5.28}$
	SP-VLA	$96.2_{5.35}$	$95.6_{5.34}$	$91.4_{5.27}$	$95.4_{5.33}$	$93.4_{5.14}$	$97.0_{5.35}$	$91.0_{5.06}$
	SparseVLM	$92.2_{6.91}$	$92.6_{7.02}$	$92.4_{6.87}$	$93.0_{6.95}$	$74.6_{6.93}$	$94.6_{6.92}$	$71.6_{6.88}$
	VLAPruner	$96.0_{6.60}$	$97.2_{6.58}$	$91.8_{6.51}$	$97.4_{6.57}$	$65.8_{6.86}$	$97.0_{6.55}$	$90.2_{6.53}$
	FastV	$96.8_{5.98}$	$98.2_{5.95}$	$91.4_{5.92}$	$96.8_{5.91}$	$91.0_{5.92}$	$97.2_{5.87}$	$91.4_{5.88}$
	VLACache	$96.6_{4.81}$	$98.2_{4.83}$	$92.6_{4.96}$	$96.6_{4.88}$	$98.2_{4.88}$	$96.6_{4.85}$	$92.6_{4.78}$
	SpecPrune-VLA	$95.4_{6.70}$	$95.8_{6.70}$	$86.2_{6.77}$	$97.0_{7.22}$	$88.0_{7.07}$	$96.0_{7.13}$	$89.0_{7.09}$
	STAR	$97.6_{7.34}$	$97.0_{7.38}$	$89.8_{7.36}$	$98.2_{6.89}$	$91.0_{7.01}$	$95.0_{6.93}$	$93.0_{6.93}$
	PyramidDrop	$92.8_{6.49}$	$92.8_{6.64}$	$84.4_{6.51}$	$97.6_{6.56}$	$94.2_{6.55}$	$96.8_{6.48}$	$89.0_{6.60}$
	AdaptPrune	$67.0_{5.53}$	$68.0_{5.49}$	$66.4_{5.57}$	$66.0_{5.52}$	$53.0_{5.41}$	$67.0_{5.47}$	$62.2_{5.49}$
10	No Pruning	$95.0_{4.74}$	$92.6_{4.73}$	$69.6_{4.71}$	$93.2_{4.74}$	$90.8_{4.71}$	$94.4_{4.68}$	$74.0_{4.71}$
	ADP	$91.0_{5.24}$	$90.0_{5.24}$	$62.4_{5.21}$	$89.8_{5.23}$	$34.0_{5.39}$	$90.6_{5.17}$	$31.0_{5.31}$
	SP-VLA	$89.8_{5.10}$	$89.6_{5.11}$	$64.0_{4.89}$	$89.8_{5.16}$	$89.8_{4.96}$	$91.0_{5.02}$	$62.2_{4.86}$
	SparseVLM	$70.6_{6.92}$	$67.6_{6.94}$	$37.2_{6.84}$	$74.0_{6.88}$	$0.6_{7.01}$	$74.2_{6.76}$	$7.8_{6.82}$
	VLAPruner	$91.8_{7.11}$	$91.2_{7.08}$	$67.4_{6.99}$	$92.2_{7.04}$	$34.0_{7.24}$	$91.6_{6.91}$	$56.2_{7.00}$
	FastV	$92.8_{5.93}$	$94.4_{5.89}$	$71.4_{5.88}$	$92.4_{5.93}$	$65.6_{5.93}$	$95.0_{5.84}$	$64.4_{5.85}$
	VLACache	$94.8_{4.83}$	$94.6_{4.81}$	$68.2_{4.76}$	$94.8_{4.88}$	$94.4_{4.82}$	$94.8_{4.79}$	$68.2_{4.83}$
	SpecPrune-VLA	$86.2_{6.87}$	$89.0_{7.29}$	$59.6_{7.06}$	$93.4_{7.41}$	$5.0_{7.53}$	$89.4_{7.30}$	$28.2_{7.23}$
	STAR	$95.0_{7.45}$	$93.2_{7.40}$	$69.6_{7.43}$	$92.8_{6.97}$	$51.0_{7.15}$	$95.0_{7.06}$	$65.0_{7.27}$
	PyramidDrop	$70.4_{6.63}$	$73.0_{6.64}$	$45.2_{6.72}$	$73.0_{6.55}$	$69.0_{6.67}$	$68.2_{6.42}$	$63.0_{6.49}$
	AdaptPrune	$31.0_{5.48}$	$26.0_{5.52}$	$28.4_{5.45}$	$31.8_{5.55}$	$10.6_{5.42}$	$22.8_{5.48}$	$14.0_{5.46}$

Finally, some pruning methods remain effective under FocusVLA. We attribute this to overfitting in OpenVLA-OFT [38, 9]. Since the model has seen the initial object layouts during training, it may still locate the target using memorized spatial cues. This overfitting is weaker in $\pi_{0.5}$, where FocusVLA shows a stronger attack effect, as shown in Appendix E. The pruning-ratio analysis in Appendix H provides further supporting evidence.

7.4 Ablation Study

To verify the necessity of each design choice, we run an ablation study on OpenVLA-OFT with LIBERO-10 under the white-patch trigger. We compare the full FocusVLA with two variants. The first replaces low-impact head selection with random head selection. The second removes clean attention preservation. We report clean no-pruning success rate (NP-SR), clean CDR (CDR-C), and triggered CDR (CDR-T).

Table 3 shows two different failure modes.**First**, without low-impact head selection, NP-SR drops from 94.4% to 81.6%. This means that randomly selected heads cannot safely carry the backdoor. We find that the drop mainly comes from tasks 1 and 2 in LIBERO-10, where the model sometimes executes the wrong instruction. This suggests that some random heads are important for instruction-task alignment. Modifying them can change the task the model tries to perform. **Second**, without clean attention preservation, NP-SR remains high, but clean CDR rises from 49.35% to 85.61%. This means that the attack pattern also appears on clean inputs. These results show that low-impact head selection preserves normal manipulation, while clean attention preservation keeps the backdoor conditional on the trigger.

Table 3: **Ablation study** on OpenVLA-OFT with LIBERO-10 and the white-patch trigger. NP-SR denotes clean success rate without pruning.

Setting	NP-SR ↑	CDR-C ↓	CDR-T ↑
Full FocusVLA	94.4	49.35	84.66
w/o Marginal Heads	81.6	50.31	85.18
w/o Clean Attn.	95.2	85.61	88.43

7.5 Persistence under Downstream Fine-tuning

To examine whether standard downstream fine-tuning can remove the backdoor, we start from the OpenVLA-OFT model poisoned on LIBERO-10 with the white-patch trigger. We then fine-tune it on clean LIBERO-Spatial data for 20k steps, without any poisoned samples or trigger-related losses. Along the fine-tuning trajectory, we evaluate each checkpoint on LIBERO-10 with the white-patch trigger. We measure final-layer CDR for both text-to-image and action-to-image attention.

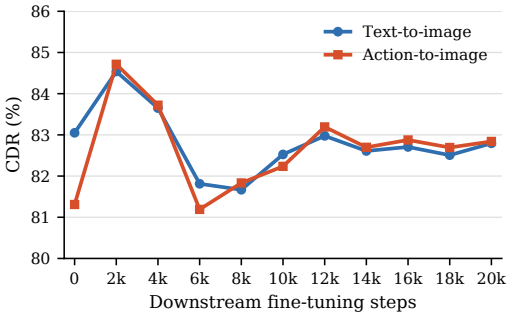

Figure 3: **Persistence of triggered attention bias under downstream fine-tuning**.

Figure 3 shows that downstream fine-tuning does not remove the trigger-conditioned attention bias. Both CDR curves stay high throughout fine-tuning and remain close to the poisoned 0k baseline. This means that the poisoned heads still push attention toward peripheral regions under the trigger, even after the model is fine-tuned on clean data from another suite. It also suggests that clean downstream fine-tuning alone may not be enough to recover normal attention behavior.

7.6 Defense via Auxiliary Token-Scoring Cues

Having demonstrated the vulnerability of attention-based pruning, we next explore how to defend against FocusVLA. Our goal is to reduce reliance on attention scores by incorporating complementary information into token selection. We evaluate ADP, FastV, SparseVLM, and VLAPruner on LIBERO-10 using OpenVLA-OFT poisoned with the white-patch trigger. All configurations retain 50% of the visual tokens, ensuring the same token budget. We consider three auxiliary scoring cues:

- **Edge saliency.** We convert each image to grayscale and apply Canny edge detection with thresholds of 100 and 200. The edge score E_i is the fraction of edge pixels in the patch corresponding to token i. This cue favors object contours and structural information.
- **Patch entropy.** We compute the normalized Shannon entropy of each grayscale patch as $H_i = -\sum_{k=0}^{255} p_{ik} \log p_{ik} / \log 256$, where p_{ik} is the fraction of pixels with intensity k. This cue favors visually diverse patches over uniform regions.
- **Spatial coverage.** We assign each token a normalized 2D coordinate and greedily select tokens. The coverage score is $C_i = \min_{j \in \mathcal{S}} d(i, j)$, where $\mathcal{S}$ is the set of selected tokens. Distances are normalized to $[0, 1]$, with a distance of 1 assigned to tokens from different cameras. We set $C_i = 1$ when $\mathcal{S}$ is empty and update the scores after each selection. This cue discourages tokens from concentrating in a small region.

For each scoring signal, we first clip its values at the 5th and 95th percentiles and then normalize them to the range $[0, 1]$. For each visual token i, we combine its attention score A_i with one auxiliary

Table 4: Defense results on LIBERO-10. Each pair reports task success rates (%) of the poisoned model on clean (C) and triggered (T) inputs. Edge, entropy, and coverage use weights of 0.3, 0.5, and 0.5, respectively. All settings retain 50% of tokens.

	Attention only		+ Edge		+ Entropy		+ Coverage	
Method	C	T	C	T	C	T	C	T
ADP	89.8	34.0	89.4	44.0	89.8	66.4	87.4	74.0
FastV	92.4	65.6	92.2	80.8	92.0	72.0	88.0	84.6
SparseVLM	74.0	0.6	71.4	68.0	71.8	58.0	73.8	65.0
VLAPruner	92.2	34.0	92.8	78.2	92.8	76.2	88.6	70.0
Average	87.1	33.6	86.5	67.8	86.6	68.2	84.5	73.4

score X_i using a weighted sum:

$$S_i = (1 - w)A_i + wX_i, \qquad X_i \in \{E_i, H_i, C_i\}. \tag{1}$$

Here, w controls the contribution of the auxiliary cue. We evaluate $w \in \{0, 0.1, 0.3, 0.5\}$, where $w = 0$ corresponds to the original attention-only scoring rule. Table 4 reports task success rates on clean and triggered inputs for three representative configurations. A higher success rate under triggered inputs indicates more effective mitigation of the attack.

First, all three auxiliary scoring cues consistently improve triggered performance across the four evaluated pruning methods. Edge saliency increases average triggered success from 33.6% to 67.8%, while clean success decreases by 0.6 percentage points. Patch entropy achieves a similar trade-off, with 68.2% triggered success and 86.6% clean success. **Second**, spatial coverage achieves the highest average triggered success among the reported defense configurations, reaching 73.4%, while clean success decreases by 2.6 points. The benefits vary across methods: edge saliency raises SparseVLM's triggered success from 0.6% to 68.0%, whereas spatial coverage provides the largest improvement for ADP among these configurations. **Third**, the auxiliary cues preserve most of the inference speed benefit. Averaged across the four methods, edge saliency, patch entropy, and spatial coverage achieve 6.03, 6.18, and 6.21 FPS, respectively, compared with 6.27 FPS for attention-only pruning and 4.73 FPS without pruning. These results show that complementary cues can mitigate the evaluated fixed attack with a moderate effect on clean performance and inference speed; robustness against an adaptive attacker remains untested.

8 Limitation

This work has two main limitations. First, FocusVLA targets attention-based visual token pruning, so it may not directly apply to methods that use other importance signals. Thus a possible defense is to avoid relying only on attention scores and combine them with other cues, such as token similarity or feature distance. Second, FocusVLA uses a spatial prior that task-relevant regions are often near the image center. This prior may not hold for all camera views, task layouts, or robot embodiments. Even so, our results show that attention can be manipulated by an attacker, which raises broader concerns for attention-based methods beyond pruning, including model inspection [25] and backdoor detection [15].

9 Conclusion

In this paper, we present FocusVLA, a supply-chain backdoor attack against attention-based visual token pruning in VLA models. FocusVLA implants trigger-conditioned attention patterns into marginal attention heads, allowing the model to behave normally without pruning or without the trigger, while misleading pruning once the trigger appears. Experiments on OpenVLA-OFT and $\pi_{0.5}$ show that the attack preserves benign performance, degrades task success under triggered pruning, and can persist after downstream fine-tuning. These results highlight that attention scores are not inherently reliable signals for safe model acceleration.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (Grant No. 62673433) and XtalPi.

References

[1] Saeed Ranjbar Alvar, Gursimran Singh, Mohammad Akbari, and Yong Zhang. Divprune: Diversity-based visual token pruning for large multimodal models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 9392–9401, 2025.

[2] Changwoo Baek, Jouwon Song, Sohyeon Kim, and Kyeongbo Kong. Agilepruner: An empirical study of attention and diversity for adaptive visual token pruning in large vision-language models. *arXiv preprint arXiv:2603.01236*, 2026.

[3] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. pi_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[4] Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *European Conference on Computer Vision*, pages 19–35. Springer, 2024.

[5] Jiawei Guo, Feifei Zhai, Pu Jian, Qianrun Wei, and Yu Zhou. Crop: Contextual region-oriented visual token pruning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 9767–9783, 2025.

[6] Yichen Guo, Hanze Li, Zonghao Zhang, Jinhao You, Kai Tang, and Xiande Huang. Star: Stage-wise attention-guided token reduction for efficient large vision-language models inference. *arXiv preprint arXiv:2505.12359*, 2025.

[7] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

[8] Sarthak Jain and Byron C Wallace. Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, 2019.

[9] Chanyoung Kim, Minwoo Kim, Minseok Kang, Hyunwoo Kim, and Dahuin Jung. Libero-para: A diagnostic benchmark and metrics for paraphrase robustness in vla models. *arXiv preprint arXiv:2603.28301*, 2026.

[10] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[11] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

[12] Duo Li, Zuhao Yang, Xiaoqin Zhang, Ling Shao, and Shijian Lu. Todre: Effective visual token pruning via token diversity and task relevance. *arXiv preprint arXiv:2505.18757*, 2025.

[13] Jiayu Li, Yunhan Zhao, Xiang Zheng, Zonghuan Xu, Yige Li, Xingjun Ma, and Yu-Gang Jiang. Attackvla: Benchmarking adversarial and backdoor attacks on vision-language-action models. *arXiv preprint arXiv:2511.12149*, 2025.

[14] Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, et al. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024.

[15] Xuetao Li, Pinhan Fu, Wenke Huang, Nengyuan Pan, Songhua Yang, Kaiyan Zhao, Guancheng Wan, Mengde Li, Jifeng Xuan, and Miao Li. When attention betrays: Erasing backdoor attacks in robotic policies by reconstructing visual tokens. *arXiv preprint arXiv:2602.03153*, 2026.

[16] Yanshu Li, Jianjiang Yang, Zhennan Shen, Ligong Han, Haoyan Xu, and Ruixiang Tang. Catp: Contextually adaptive token pruning for efficient and enhanced multimodal in-context learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 6619–6627, 2026.

[17] Ye Li, Yuan Meng, Zewen Sun, Kangye Ji, Chen Tang, Jiajun Fan, Xinzhu Ma, Shutao Xia, Zhi Wang, and Wenwu Zhu. Sp-vla: A joint model scheduling and token pruning approach for vla model acceleration. *arXiv preprint arXiv:2506.12723*, 2025.

[18] Zhixuan Liang, Yizhuo Li, Tianshuo Yang, Chengyue Wu, Sitong Mao, Tian Nian, Liuao Pei, Shunbo Zhou, Xiaokang Yang, Jiangmiao Pang, et al. Discrete diffusion vla: Bringing discrete diffusion to action decoding in vision-language-action policies. *arXiv preprint arXiv:2508.20072*, 2025.

[19] Ziyan Liu, Yeqiu Chen, Hongyi Cai, Tao Lin, Shuo Yang, Zheng Liu, and Bo Zhao. Vla-pruner: Temporal-aware dual-level visual token pruning for efficient vision-language-action inference. *arXiv preprint arXiv:2511.16449*, 2025.

[20] Bozhi Luan, Wengang Zhou, Hao Feng, Zhe Wang, Xiaosong Li, and Houqiang Li. Multi-cue adaptive visual token pruning for large vision-language models. *arXiv preprint arXiv:2503.08019*, 2025.

[21] Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32, 2019.

[22] Xiaohuan Pei, Yuxing Chen, Siyu Xu, Yunke Wang, Yuheng Shi, and Chang Xu. Action-aware dynamic pruning for efficient vision-language-action manipulation. *arXiv preprint arXiv:2509.22093*, 2025.

[23] Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

[24] Wenxuan Song, Jiayi Chen, Pengxiang Ding, Han Zhao, Wei Zhao, Zhide Zhong, Zongyuan Ge, Jun Ma, and Haoang Li. Accelerating vision-language-action model integrated with action chunking via parallel decoding. *arXiv preprint arXiv:2503.02310*, 2025.

[25] Jesse Vig. A multiscale visualization of attention in the transformer model. In *Proceedings of the 57th annual meeting of the association for computational linguistics: system demonstrations*, pages 37–42, 2019.

[26] Hanzhen Wang, Jiaming Xu, Yushun Xiang, Jiayi Pan, Yongkang Zhou, Yong-Lu Li, and Guohao Dai. Specprune-vla: Accelerating vision-language-action models via action-aware self-speculative pruning. *arXiv preprint arXiv:2509.05614*, 2025.

[27] Xin Wang, Jie Li, Zejia Weng, Yixu Wang, Yifeng Gao, Tianyu Pang, Chao Du, Yan Teng, Yingchun Wang, Zuxuan Wu, et al. Freezevla: Action-freezing attacks against vision-language-action models. *arXiv preprint arXiv:2509.19870*, 2025.

[28] Long Xing, Qidong Huang, Xiaoyi Dong, Jiajie Lu, Pan Zhang, Yuhang Zang, Yuhang Cao, Conghui He, Jiaqi Wang, Feng Wu, et al. Pyramiddrop: Accelerating your large vision-language models via pyramid visual redundancy reduction. *arXiv preprint arXiv:2410.17247*, 2024.

[29] Siyu Xu, Yunke Wang, Chenghao Xia, Dihao Zhu, Tao Huang, and Chang Xu. Vla-cache: Efficient vision-language-action manipulation via adaptive token caching. *arXiv preprint arXiv:2502.02175*, 2025.

[30] Zonghuan Xu, Xiang Zheng, Xingjun Ma, and Yu-Gang Jiang. Tabvla: Targeted backdoor attacks on vision-language-action models. *arXiv preprint arXiv:2510.10932*, 2025.

[31] Sihan Yang, Runsen Xu, Chenhang Cui, Tai Wang, Dahua Lin, and Jiangmiao Pang. Vflowopt: A token pruning framework for lmms with visual information flow-guided optimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23924–23934, 2025.

[32] Yantai Yang, Yuhao Wang, Zichen Wen, Luo Zhongwei, Chang Zou, Zhipeng Zhang, Chuan Wen, and Linfeng Zhang. Efficientvla: Training-free acceleration and compression for vision-language-action models. *arXiv preprint arXiv:2506.10100*, 2025.

[33] Boqiang Zhang, Lei Ke, Ruihan Yang, Qi Gao, Tianyuan Qu, Rossell Chen, Dong Yu, et al. Penguin-vl: Exploring the efficiency limits of vlm with llm-based vision encoders. *arXiv preprint arXiv:2603.06569*, 2026.

[34] Qizhe Zhang, Aosong Cheng, Ming Lu, Renrui Zhang, Zhiyong Zhuo, Jiajun Cao, Shaobo Guo, Qi She, and Shanghang Zhang. Beyond text-visual attention: Exploiting visual cues for effective token pruning in vlms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20857–20867, 2025.

[35] Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, et al. Sparsevlm: Visual token sparsification for efficient vision-language model inference. *arXiv preprint arXiv:2410.04417*, 2024.

[36] Jianyi Zhou, Yujie Wei, Ruichen Zhen, Bo Zhao, Xiaobo Xia, Rui Shao, Xiu Su, and Shuo Yang. Inject once survive later: Backdooring vision-language-action models to persist through downstream fine-tuning. *arXiv preprint arXiv:2602.00500*, 2026.

[37] Xueyang Zhou, Guiyao Tie, Guowen Zhang, Hechang Wang, Pan Zhou, and Lichao Sun. Badvla: Towards backdoor attacks on vision-language-action models via objective-decoupled optimization. *arXiv preprint arXiv:2505.16640*, 2025.

[38] Xueyang Zhou, Yangming Xu, Guiyao Tie, Yongchao Chen, Guowen Zhang, Duanfeng Chu, Pan Zhou, and Lichao Sun. Libero-pro: Towards robust and fair evaluation of vision-language-action models beyond memorization. *arXiv preprint arXiv:2510.03827*, 2025.

[39] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023.

Appendix

A Additional Motivational Results

This appendix provides additional analyses supporting the motivational study in Section 4.1 and Section 4.2. The main paper reports the most representative heatmaps for OpenVLA-OFT. Here, we include complementary layer-wise statistics and an entropy analysis that helps explain the behavior of early layers.

A.1 Layer-Wise Head Divergence and Entropy

Figure 4 left summarizes the layer-wise trend behind attention-head heterogeneity. The mean pairwise JS divergence is small in shallow layers and gradually increases in deeper layers. This suggests that, as representations become more task-specific, attention heads also become more diverse.

However, the low divergence in shallow layers does not mean that different heads extract the same task-specific information. Figure 4 middle and right show the attention entropy of different layers in OpenVLA-OFT, and its correlation with JS divergence. We observe that shallow-layer attention has high entropy and is close to a uniform distribution over visual tokens. When all heads broadly attend to the whole image, their pairwise divergence is naturally small. In deeper layers, attention becomes more selective, and the divergence between heads increases. Overall, layer-wise attention entropy shows a strong negative correlation with JS divergence.

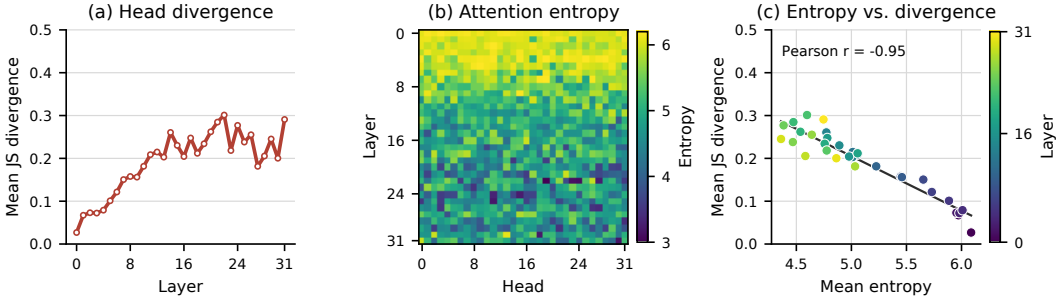

Figure 4: **Additional analysis for head heterogeneity in OpenVLA-OFT**. (a) Mean pairwise JS divergence between attention heads across layers. (b) Attention entropy across layers and heads. (c) Relationship between layer-wise mean entropy and layer-wise mean JS divergence.

A.2 Layer-Wise Distribution of Redundant Heads

Figure 5 reports the number of heads in each OpenVLA-OFT layer whose single-head ablation lowers the action loss. Such heads appear across most layers, suggesting that redundancy is not limited to a small part of the model.

A.3 Motivational Results on $\pi_{0.5}$

We also repeat the motivational analysis on $\pi_{0.5}$. Figure 6 shows the final-layer pairwise JS divergence between attention heads. Both text-to-image and action-to-image attention exhibit clear head-level diversity, consistent with the OpenVLA-OFT results in the main paper.

Figure 7 reports the single-head ablation results for the VLM and action expert in $\pi_{0.5}$. Many heads have small or negative impact on the action loss, again indicating substantial head redundancy.

B Stability of First-Order Head Importance

In the motivational study, we use single-head ablation to show that many attention heads contribute little to manipulation. This ablation-based analysis is useful for diagnosis, but it is too expensive for attack construction because it requires a separate forward pass for each head. For example,

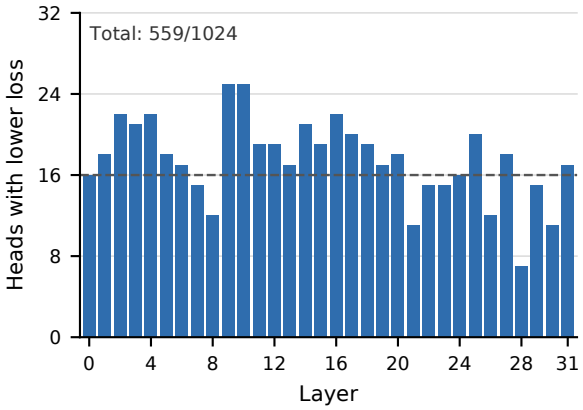

Figure 5: Heads per layer whose ablation reduces action loss in OpenVLA-OFT.

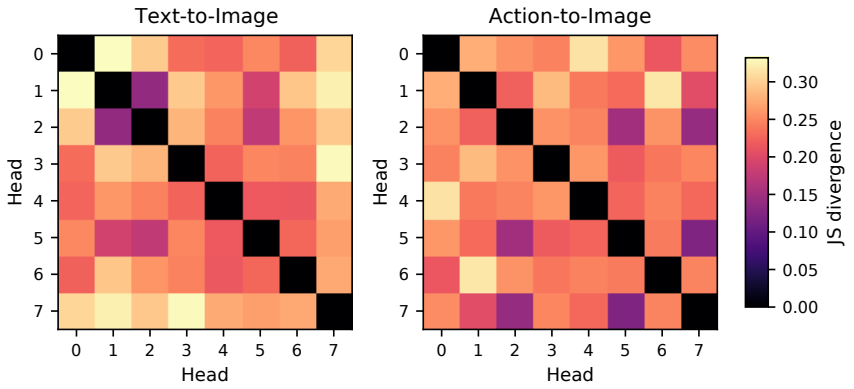

Figure 6: Pairwise JS divergence between attention heads in the final layer of $\pi_{0.5}$.

OpenVLA-OFT has 1,024 attention heads. Evaluating only 100 samples would require 102,400 forward passes, and 100 samples are still not enough to select heads reliably. Therefore, in our main method, we use the first-order importance score in Section 6 to identify marginal heads more efficiently.

We further verify that this first-order criterion gives stable head rankings across different downstream task suites. We compute the importance scores of all OpenVLA-OFT attention heads on four LIBERO suites: Spatial, Object, Goal, and LIBERO-10. For each Transformer layer, each suite produces a 32-dimensional importance vector over the heads in that layer. We then compute the Spearman rank correlation between every pair of suites and average the six pairwise correlations. This measures whether different suites induce similar head-importance rankings within the same layer.

Figure 8 shows the layer-wise results. The mean Spearman correlation across all layers and suite pairs is 0.821, and the median is 0.845. This indicates that the relative importance of attention heads is largely consistent across different LIBERO suites. Therefore, the marginal heads selected by the first-order criterion are not specific to a single task suite, supporting its use as an efficient alternative to exhaustive single-head ablation in our attack construction.

C Training and Acceleration Details

Hardware. We conduct all experiments on a local server equipped with two Intel Xeon Gold 6226R CPUs and 8 NVIDIA GeForce RTX 3090 GPUs, each with 24GB of memory. The server provides 64 CPU threads in total.

OpenVLA-OFT training details. For OpenVLA-OFT, we start from the official suite-specific LIBERO checkpoints and further poison each checkpoint with LoRA fine-tuning. We train a separate

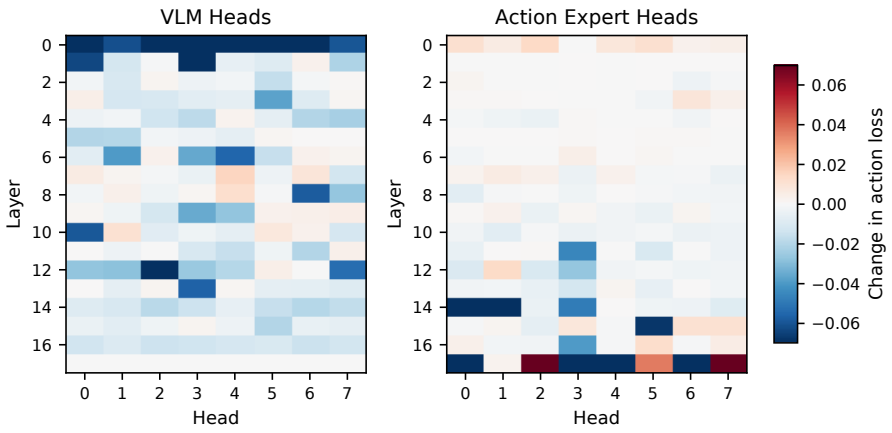

Figure 7: Single-head ablation results in $\pi_{0.5}$. Negative values indicate that ablating the head lowers the action loss.

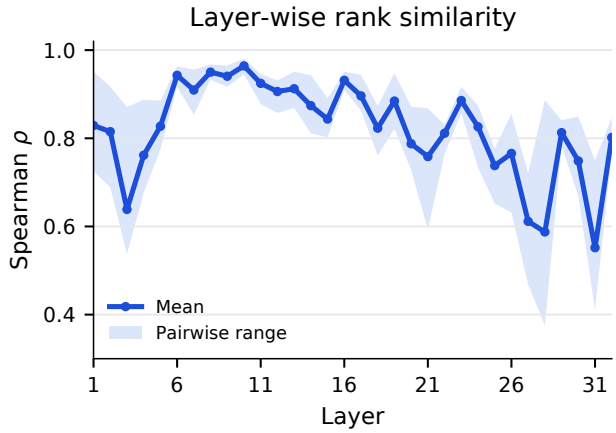

Figure 8: Layer-wise stability of first-order head importance in OpenVLA-OFT across four LIBERO suites. For each layer, we compute the Spearman rank correlation of the 32 head-importance scores between every pair of suites. The curve shows the pairwise mean, and the shaded region shows the pairwise range.

poisoned model for each LIBERO suite and each trigger, with 1,000 training steps for each run. We apply LoRA to all linear layers with rank 32 and dropout 0. The model takes two camera views as input and uses proprioceptive states. We train with batch size 2 for clean samples and batch size 2 for triggered samples, gradient accumulation 4, AdamW optimizer, and learning rate 5×10^{-4}. Each update optimizes the same objective as in Section 6, including $\mathcal{L}_{\text{attack}}$, $\mathcal{L}_{\text{act}}^{\text{tri}}$, $\mathcal{L}_{\text{act}}^{\text{clean}}$, and $\mathcal{L}_{\text{js}}$. To improve instruction generalization, we also apply the random-text triggered attention term in $\mathcal{L}_{\text{attack}}$ to poisoned samples whose non-special text tokens are randomly replaced. We set the clean attention-preservation weight to 2.0 and $\lambda_{\text{rand}} = 0.1$.

$\pi_{0.5}$ training details. For $\pi_{0.5}$, we start from the LeRobot checkpoint trained on all four LIBERO suites and poison the same model for all suites. We use LoRA fine-tuning on all linear layers with rank 32 and dropout 0. The batch sizes for clean teacher samples, triggered samples, and clean replay samples are all 4, with gradient accumulation 4. We train each poisoned model for 1,000 steps using AdamW with learning rate 1×10^{-4}, weight decay 0.001, 10 warmup steps, and a minimum learning rate of 2.5×10^{-5}. The attack is applied to both the VLM branch and the action-expert branch. We select 4 heads per layer by default (2 heads of VLM and 2 heads of action expert), and use the same random text-token replacement strategy for poisoned samples. We use weight 1.0 for $\mathcal{L}_{\text{attack}}$, $\mathcal{L}_{\text{act}}^{\text{tri}}$, and $\mathcal{L}_{\text{act}}^{\text{clean}}$ and set $\lambda_{\text{rand}} = 0.1$.

Acceleration baseline details. We evaluate ten representative acceleration methods that cover the main ways of reducing visual-token overhead in VLA systems. All methods use the same LIBERO evaluation protocol as in the main experiments. Unless otherwise specified, we follow the default settings in our local implementation and report FPS as the inverse of the average single-step prediction time. Below, we describe each pruning method and its parameter settings for OpenVLA-OFT.

- **FastV.** A simple token pruning method that keeps early visual evidence and drops redundant tokens in deeper layers. We use decoder layer 3 and prune ratio 0.5.
- **SparseVLM.** A text-guided sparsification method that uses attention scores to rank visual tokens and applies layer-wise token reduction. We prune at layers 2, 6, and 15, with target token counts 300, 200, and 110.
- **VLAPruner.** A VLA-specific pruning method that combines current attention with temporal attention. We use pruning layer 3, prune ratio 0.75, temporal layer 15, and temporal window 3.
- **ADP.** A dynamic pruning method that adapts token selection to the current manipulation stage. We use QK selection at layer 0 with keep ratio 0.5, and split the kept tokens across the two camera views with a 0.4/0.6 ratio.
- **SP.** A spatio-semantic pruning method that adjusts token selection using visual attention and image edges. We use the automatic profile, history size 6, attention thresholds $z_{\mathrm{thre}} = 0.5$ and $z_{\min} = 0.9$, and Canny edge thresholds 100/200.
- **VLACache.** A token caching method that reuses visual computation across nearby steps. We use static top-150 tokens, attention top-100 tokens, attention layer 15, and similarity threshold 0.996.
- **SpecPrune-VLA.** A self-speculative pruning method that combines static token selection with action-aware control. We use static pruning after layer 3, global scoring layers 15 and 32, $\alpha = 0.8$, and minimum static keep ratio 0.25.
- **STAR.** A spatial-temporal routing method that prunes visual tokens in two stages. We use stage-1 keep ratio 0.75 and stage-2 pruning at decoder layer 3 with keep ratio 0.5, giving a final keep ratio of 0.375.
- **PyramidDrop.** A progressive pruning method that drops redundant tokens in a coarse-to-fine manner across layers. We use pruning layers 8, 16, and 24, with keep ratios 0.5, 0.25, and 0.125.
- **AdaptPrune.** An adaptive pruning method that combines attention, spatial distance, and token similarity. We use pruning layer 2, keep ratio 0.5, distance limit 5.0, and text-last query mode.

D Layer-0 Triggered Attention Bias

Table 5 reports the layer-0 CDR results for OpenVLA-OFT and $\pi_{0.5}$. The setup follows Section 6. We compute CDR under both clean and triggered inputs, using text-to-image and action-to-image attention. The results show that, even in layer 0, $\pi_{0.5}$ already shows a clear attention bias when the trigger appears. OpenVLA-OFT shows a similar pattern for text-to-image attention. However, its action-to-image CDR stays at 51.56% in all settings. This is likely due to the model architecture of OpenVLA-OFT. At the input layer, action tokens are still placeholders and have not yet absorbed enough cross-modal information from visual and text tokens. As a result, the overlap-based CDR metric has limited discriminative power for action queries in layer 0.

To further verify that this constant value is specific to the input layer rather than a failure of the attack, we measure action-to-image CDR at every Transformer layer of OpenVLA-OFT on LIBERO-Spatial. Figure 9 shows that action-to-image CDR is close to the non-informative baseline at layer 0, but rises sharply from layer 1 and remains high in later layers for both triggers. This indicates that after even a small amount of Transformer processing, action tokens start to carry cross-modal information and reveal the trigger-conditioned attention bias. Therefore, the constant layer-0 action CDR should be interpreted as an architectural artifact of OpenVLA-OFT, not as evidence that the attack fails on action-to-image attention.

Table 5: Layer-0 triggered attention bias measured by CDR (%).

Variant	LIBERO-10			LIBERO-Goal			LIBERO-Object			LIBERO-Spatial		
	Clean	White	Cup	Clean	White	Cup	Clean	White	Cup	Clean	White	Cup
OpenVLA-OFT												
Benign_T	40.16	41.81	40.92	39.24	40.57	39.94	37.62	37.98	37.32	41.03	42.18	41.10
Benign_A	51.56	51.56	51.56	51.56	51.56	51.56	51.56	51.56	51.56	51.56	51.56	51.56
FocusVLA-W_T	40.44	**87.56**	40.69	37.23	**87.39**	37.44	48.89	**88.45**	45.90	42.23	**86.87**	42.44
FocusVLA-W_A	51.56	**51.56**	51.56	51.56	**51.56**	51.56	51.56	**51.56**	51.56	51.56	**51.56**	51.56
FocusVLA-C_T	48.17	47.87	**73.41**	39.89	41.61	**71.34**	87.42	87.52	**87.85**	43.99	45.32	**81.27**
FocusVLA-C_A	51.56	51.56	**51.56**	51.56	51.56	**51.56**	51.56	51.56	**51.56**	51.56	51.56	**51.56**
$\boldsymbol{\pi_{0.5}}$												
Benign_T	48.49	49.36	49.18	46.18	46.57	47.30	43.20	44.18	42.77	49.11	50.21	49.07
Benign_A	48.66	49.50	49.74	48.83	49.38	48.96	48.33	49.66	48.44	48.89	50.61	49.06
FocusVLA-W_T	46.47	**97.23**	46.01	43.43	**96.84**	44.39	41.30	**97.61**	41.54	48.44	**97.45**	48.66
FocusVLA-W_A	48.41	**96.50**	47.69	47.39	**96.13**	47.38	48.44	**96.44**	48.73	47.68	**96.50**	47.83
FocusVLA-C_T	50.02	50.17	**77.46**	46.71	47.10	**76.99**	44.76	45.92	**80.94**	53.90	54.60	**78.16**
FocusVLA-C_A	49.91	50.69	**77.57**	49.74	50.24	**76.58**	49.69	50.84	**80.84**	51.11	52.30	**77.01**

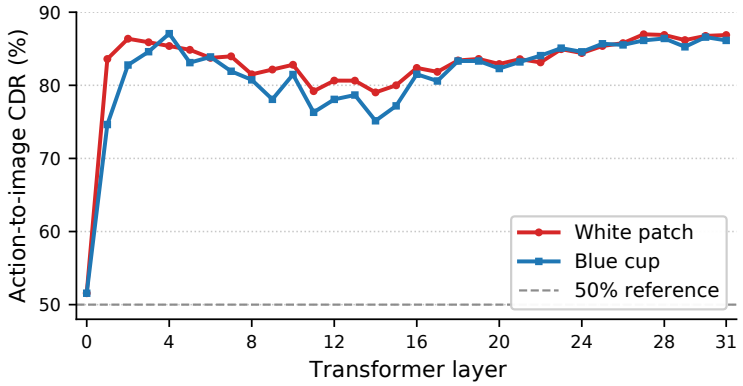

Figure 9: Layer-wise action-to-image CDR on OpenVLA-OFT with LIBERO-Spatial. Layer 0 gives a nearly non-informative CDR value, while later layers show clear trigger-conditioned attention bias.

E Pruning result on $\pi_{0.5}$

To further study whether FocusVLA is also effective on other models, we evaluated 10 pruning methods on $\pi_{0.5}$ as well. For each task, we ran 20 trials. The pruning hyperparameters were kept the same as those used for OpenVLA-OFT. The results are reported in Table 6.

First, compared with OpenVLA-OFT, $\pi_{0.5}$ is less overfitted. As a result, most pruning methods lead to a larger drop in success rate (SR) on $\pi_{0.5}$ than on OpenVLA-OFT. **Second**, FocusVLA achieves a much stronger attack effect on $\pi_{0.5}$ than on OpenVLA-OFT. Across all LIBERO scenarios, most pruning methods show a clear SR drop when the backdoor is activated. **Third**, FocusVLA-W performs better than FocusVLA-C, which is consistent with the results on OpenVLA-OFT.

F Effect of Controlled Heads

To study how the number of controlled attention heads affects the attack, we vary the number of heads controlled in each layer from 4 to 24 on OpenVLA-OFT with LIBERO-10. We evaluate the poisoned models under the white-patch trigger and report the clean success rate without pruning together with text-to-image CDR (on the final layer) under the triggered input. Figure 10 summarizes the trade-off.

Figure 10 shows two clear trends. First, controlling too many heads can hurt the model's normal manipulation ability. When the number of controlled heads reaches 20, the clean success rate drops sharply. A closer inspection shows the same failure pattern as in the random-head ablation: the

Table 6: **Purning results on $\pi_{0.5}$ across four LIBERO suites**. Each cell reports SR, with FPS shown as a blue subscript.

Suite	Method	Benign Model			FocusVLA-W		FocusVLA-C	
		Clean	White	Cup	Clean	White	Clean	Cup
Spatial	No Pruning	$97.0_{2.70}$	$98.0_{2.77}$	$96.5_{2.68}$	$96.5_{2.62}$	$93.5_{2.71}$	$98.5_{2.67}$	$95.0_{2.68}$
	ADP	$88.5_{3.13}$	$92.0_{3.10}$	$87.5_{3.13}$	$91.5_{3.15}$	$27.0_{3.10}$	$88.0_{3.08}$	$53.0_{3.13}$
	SP-VLA	$78.5_{3.10}$	$77.5_{3.14}$	$78.5_{3.19}$	$85.0_{3.14}$	$43.5_{3.13}$	$82.5_{3.16}$	$76.5_{3.16}$
	SparseVLM	$95.5_{3.27}$	$95.5_{3.25}$	$81.5_{3.20}$	$92.0_{3.20}$	$0.0_{3.25}$	$81.0_{3.19}$	$14.0_{3.21}$
	VLAPruner	$91.0_{2.95}$	$93.5_{2.98}$	$85.0_{2.95}$	$94.5_{2.93}$	$1.0_{2.95}$	$88.0_{2.99}$	$44.0_{2.94}$
	FastV	$91.0_{3.02}$	$92.5_{3.01}$	$86.5_{3.11}$	$91.0_{3.12}$	$0.0_{3.10}$	$66.5_{3.08}$	$29.0_{3.09}$
	VLACache	$92.5_{2.89}$	$93.5_{2.96}$	$89.0_{2.96}$	$94.0_{2.97}$	$80.5_{3.07}$	$95.0_{2.95}$	$95.0_{3.07}$
	SpecPrune-VLA	$94.0_{3.19}$	$95.5_{3.11}$	$92.0_{3.14}$	$92.0_{3.13}$	$76.0_{3.13}$	$96.0_{3.07}$	$87.0_{3.06}$
	STAR	$82.5_{3.18}$	$86.0_{3.12}$	$80.5_{3.08}$	$81.0_{3.10}$	$63.0_{3.09}$	$93.0_{3.12}$	$93.0_{3.13}$
	PyramidDrop	$89.5_{2.96}$	$86.5_{2.97}$	$86.0_{2.93}$	$71.0_{2.94}$	$58.0_{2.96}$	$87.0_{2.95}$	$87.0_{2.95}$
	AdaptPrune	$65.0_{3.09}$	$69.0_{3.10}$	$65.5_{3.09}$	$70.0_{3.06}$	$32.0_{3.10}$	$69.0_{3.12}$	$42.0_{3.03}$
Object	No Pruning	$99.0_{2.76}$	$100.0_{2.66}$	$92.0_{2.74}$	$100.0_{2.46}$	$96.0_{2.72}$	$99.0_{2.71}$	$90.5_{2.61}$
	ADP	$95.0_{3.12}$	$98.0_{3.01}$	$92.5_{3.11}$	$98.0_{3.13}$	$30.5_{3.08}$	$98.0_{3.12}$	$63.0_{3.14}$
	SP-VLA	$96.5_{3.07}$	$97.5_{2.99}$	$90.5_{3.01}$	$93.0_{2.96}$	$61.0_{2.96}$	$98.0_{3.07}$	$82.0_{3.02}$
	SparseVLM	$99.0_{3.18}$	$99.0_{3.08}$	$93.5_{3.18}$	$98.5_{3.12}$	$0.0_{3.03}$	$99.0_{3.05}$	$8.0_{3.12}$
	VLAPruner	$97.0_{3.04}$	$98.5_{3.12}$	$92.0_{1.93}$	$100.0_{3.11}$	$1.0_{3.07}$	$98.0_{2.98}$	$34.0_{2.95}$
	FastV	$98.0_{3.04}$	$99.0_{2.93}$	$91.0_{3.09}$	$99.0_{2.99}$	$0.0_{3.05}$	$97.0_{3.01}$	$35.0_{2.97}$
	VLACache	$97.0_{2.98}$	$97.5_{2.97}$	$93.5_{3.09}$	$99.5_{2.94}$	$83.0_{3.06}$	$99.0_{2.99}$	$82.0_{2.93}$
	SpecPrune-VLA	$99.0_{3.26}$	$99.5_{3.09}$	$92.5_{3.04}$	$99.0_{3.11}$	$84.0_{3.15}$	$98.0_{3.24}$	$89.0_{3.24}$
	STAR	$94.0_{3.24}$	$94.0_{3.06}$	$89.5_{3.10}$	$95.0_{3.17}$	$79.0_{3.16}$	$97.0_{3.16}$	$88.0_{3.06}$
	PyramidDrop	$95.0_{3.00}$	$96.0_{2.88}$	$87.5_{2.91}$	$96.0_{2.84}$	$78.0_{2.97}$	$96.0_{2.93}$	$85.0_{2.96}$
	AdaptPrune	$75.0_{3.19}$	$78.0_{3.10}$	$61.0_{2.97}$	$76.0_{3.14}$	$18.0_{3.02}$	$74.0_{3.05}$	$48.0_{3.11}$
Goal	No Pruning	$97.0_{2.76}$	$97.0_{2.70}$	$94.5_{2.78}$	$96.0_{2.78}$	$94.5_{2.63}$	$97.0_{2.70}$	$95.5_{2.70}$
	ADP	$92.0_{3.17}$	$92.0_{3.11}$	$89.0_{3.11}$	$92.5_{3.15}$	$34.0_{3.06}$	$94.0_{3.12}$	$58.0_{3.10}$
	SP-VLA	$83.0_{3.14}$	$88.0_{3.11}$	$85.0_{3.10}$	$86.5_{3.09}$	$43.0_{3.17}$	$85.5_{3.08}$	$70.0_{3.15}$
	SparseVLM	$91.0_{3.26}$	$91.0_{3.20}$	$79.5_{3.23}$	$85.0_{3.29}$	$4.5_{3.18}$	$90.0_{3.25}$	$15.0_{3.17}$
	VLAPruner	$93.5_{2.96}$	$93.5_{3.08}$	$90.5_{2.95}$	$96.5_{3.04}$	$3.5_{2.90}$	$87.0_{3.03}$	$30.0_{3.01}$
	FastV	$90.5_{3.11}$	$92.0_{3.02}$	$90.5_{3.05}$	$97.0_{3.07}$	$3.5_{3.08}$	$79.0_{3.18}$	$29.5_{3.05}$
	VLACache	$89.5_{3.05}$	$90.0_{2.97}$	$84.5_{2.99}$	$92.0_{3.03}$	$66.5_{2.95}$	$96.0_{2.98}$	$84.0_{3.05}$
	SpecPrune-VLA	$95.0_{3.18}$	$94.5_{3.22}$	$89.5_{3.27}$	$97.0_{3.16}$	$77.0_{3.24}$	$93.5_{3.20}$	$79.0_{3.10}$
	STAR	$94.0_{3.23}$	$91.5_{3.20}$	$94.0_{3.22}$	$95.0_{3.24}$	$68.0_{3.22}$	$92.0_{3.17}$	$86.0_{3.23}$
	PyramidDrop	$87.5_{3.04}$	$90.0_{3.05}$	$85.5_{3.03}$	$91.0_{3.07}$	$69.0_{3.09}$	$89.5_{2.94}$	$82.0_{3.05}$
	AdaptPrune	$78.5_{3.16}$	$79.5_{3.13}$	$72.5_{3.17}$	$76.0_{3.11}$	$31.0_{3.14}$	$79.0_{3.13}$	$62.0_{3.05}$
10	No Pruning	$95.5_{2.69}$	$93.5_{2.68}$	$77.0_{2.75}$	$94.5_{2.63}$	$91.0_{2.74}$	$93.5_{2.69}$	$72.0_{2.67}$
	ADP	$79.5_{3.12}$	$80.5_{3.07}$	$66.0_{3.11}$	$81.5_{3.14}$	$7.0_{3.12}$	$79.0_{3.13}$	$37.0_{3.06}$
	SP-VLA	$84.0_{3.04}$	$82.5_{2.99}$	$71.0_{3.01}$	$82.5_{3.03}$	$40.0_{3.02}$	$81.0_{2.96}$	$63.0_{3.05}$
	SparseVLM	$93.0_{3.17}$	$95.0_{3.09}$	$74.5_{3.11}$	$92.5_{3.11}$	$0.0_{3.04}$	$87.0_{3.14}$	$7.0_{3.11}$
	VLAPruner	$91.5_{2.96}$	$88.0_{3.05}$	$71.5_{3.04}$	$90.5_{2.89}$	$0.0_{2.82}$	$94.0_{2.96}$	$20.0_{2.84}$
	FastV	$86.0_{2.98}$	$83.0_{3.04}$	$62.5_{3.05}$	$91.0_{3.05}$	$0.0_{2.95}$	$88.0_{3.08}$	$14.0_{2.97}$
	VLACache	$91.0_{2.90}$	$90.5_{2.81}$	$71.0_{2.99}$	$95.0_{2.83}$	$53.5_{2.73}$	$96.0_{2.92}$	$65.0_{2.88}$
	SpecPrune-VLA	$94.0_{3.24}$	$90.5_{3.18}$	$68.0_{3.17}$	$91.5_{3.24}$	$75.0_{3.16}$	$93.0_{3.28}$	$73.0_{3.23}$
	STAR	$86.0_{3.15}$	$89.5_{3.15}$	$66.0_{3.17}$	$82.5_{3.17}$	$52.0_{3.18}$	$91.0_{3.18}$	$72.0_{3.04}$
	PyramidDrop	$84.5_{2.91}$	$80.5_{2.97}$	$63.5_{2.93}$	$83.0_{2.95}$	$48.0_{2.98}$	$81.0_{2.83}$	$67.0_{2.87}$
	AdaptPrune	$47.5_{3.22}$	$45.5_{3.14}$	$37.0_{3.28}$	$41.0_{3.15}$	$2.0_{3.15}$	$58.0_{3.24}$	$28.0_{3.16}$

model often confuses tasks 1 and 2 in LIBERO-10, suggesting that excessive head manipulation can disrupt the information needed to distinguish similar language instructions. Second, the attack effect becomes stable once 8 heads are controlled in each layer. The text-to-image CDR remains around 0.84 from 8 heads onward, indicating that further increasing the number of controlled heads brings little additional gain. Considering both clean success rate and triggered CDR, we use 8 controlled heads per layer for all OpenVLA-OFT experiments.

G Cross-Suite Transferability

To examine whether the learned attention bias transfers across task suites, we train the poisoned OpenVLA-OFT model on one LIBERO suite with the white-patch trigger and evaluate the resulting model on all four LIBERO suites. We measure final-layer CDR without applying pruning, since the goal of this experiment is to isolate the transferability of the trigger-conditioned attention pattern

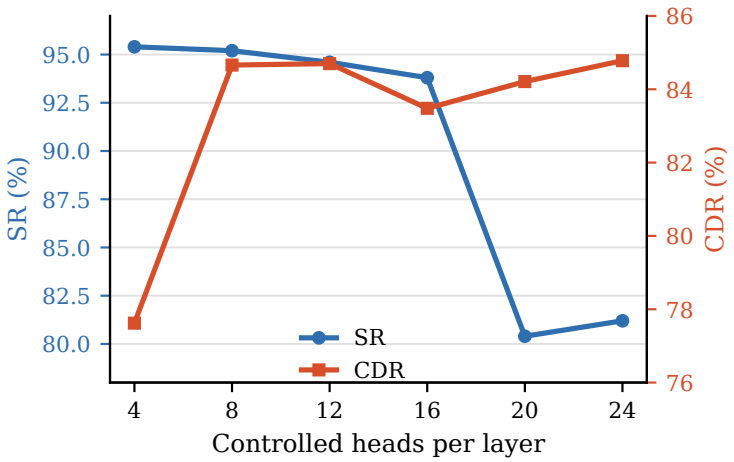

Figure 10: Effect of the number of controlled heads per layer on OpenVLA-OFT with LIBERO-10 and the white-patch trigger. SR is measured without pruning, and CDR is computed from text-to-image attention under the triggered input.

Table 7: Cross-suite transferability of FocusVLA on OpenVLA-OFT with the white-patch trigger. Rows indicate the source suite used for poisoning, and columns indicate the target suite used for evaluation. We report text-to-image and action-to-image CDR (%) separately.

Text-to-Image CDR (%)				
Source → Target	Spatial	Object	Goal	LIBERO-10
Spatial	86.95	87.73	87.26	87.13
Object	83.17	84.21	83.63	83.51
Goal	85.83	85.36	84.71	85.81
LIBERO-10	85.02	84.58	84.22	84.66
Action-to-Image CDR (%)				
Source → Target	Spatial	Object	Goal	LIBERO-10
Spatial	86.88	87.34	87.06	86.88
Object	84.18	85.69	85.15	84.84
Goal	85.59	85.90	84.88	85.97
LIBERO-10	85.19	84.32	84.00	84.76

itself. We report both text-to-image and action-to-image CDR, which allows us to check whether the learned bias generalizes across different query modalities.

Table 7 reports the results. The CDR remains consistently high not only on the source suite used for poisoning, but also on the unseen target suites. This shows that the learned spatial attention bias is not tied to a specific LIBERO scene distribution or a fixed set of task instructions. In other words, even when the visual environment and language prompts change, FocusVLA can still induce the same trigger-conditioned attention shift. This supports the transferability of the attack across downstream task suites.

H Effect of Pruning Ratio

To study how pruning strength affects FocusVLA, we evaluate OpenVLA-OFT on LIBERO-Spatial with the white-patch trigger under different pruning ratios. We choose LIBERO-Spatial because harder suites such as LIBERO-10 show a rapid accuracy drop after pruning, often close to zero, which

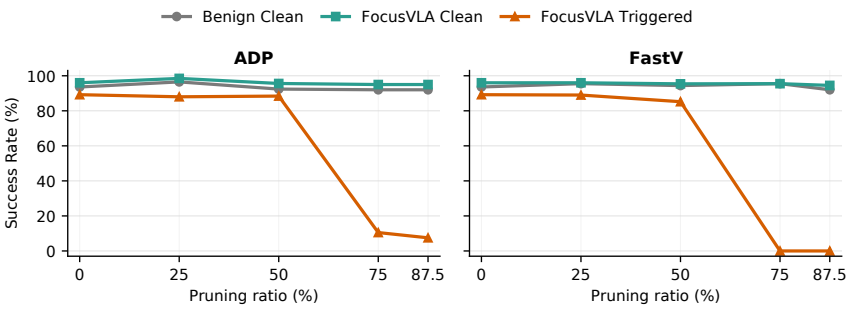

Figure 11: **Effect of pruning ratio on OpenVLA-OFT with LIBERO-Spatial and the white-patch trigger**. FocusVLA preserves clean performance across pruning ratios, while triggered performance drops sharply when the pruning ratio becomes high.

leaves little room for analysis. We use ADP and FastV as two representative pruning methods. A pruning ratio of 0% corresponds to no pruning, and 50% corresponds to the default setting used in the main experiments. For each ratio, we compare the benign model on clean inputs, the poisoned model on clean inputs, and the poisoned model on triggered inputs.

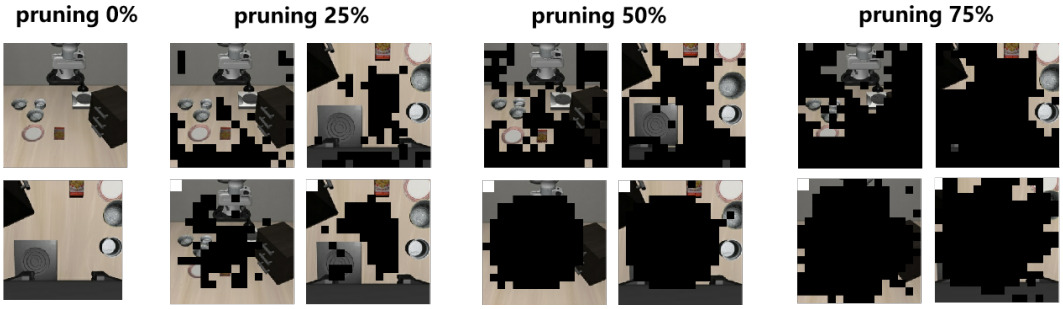

Figure 12: **Case Study of FocusVLA-W under ADP Pruning on LIBERO-Spatial**. The first row shows clean inputs, and the second row shows white-patch inputs under different pruning ratios.

Figure 12 shows the case study and Figure 11 shows three findings. **First**, under all pruning ratios, the poisoned model on clean inputs stays close to the clean model. This means that stronger pruning does not expose the backdoor in benign settings. **Second**, model performance under triggered inputs is much more sensitive to the pruning ratio. When pruning is mild or moderate, the model can still keep enough task-relevant visual information, so the triggered success rate only drops slightly. When the pruning ratio reaches 75% or higher, both ADP and FastV discard most visual tokens. In this case, the trigger-induced attention shift is enough to remove task-critical center regions, and the triggered success rate drops sharply. The drop is especially clear for FastV. **Third**, even when the benign model prunes 87.5% of visual tokens, ADP and FastV still keep over 90% SR on LIBERO-Spatial. This supports our earlier hypothesis that OpenVLA-OFT is heavily overfitted. As long as the model observes a few object positions in the image, it may act based on memorized training layouts. This helps explain why FocusVLA is less effective under some OpenVLA-OFT settings.

I Real-World Experiments

Platform and model. We conduct the real-robot study on the Astribot S1 humanoid robot from Stardust Intelligence (Astribot). We use $\pi_{0.5}$ for this evaluation. We use only the head and right-wrist cameras as inputs. All tasks are executed with the robot's right arm only, while the base and waist remain fixed. We use a white cup as the physical trigger.

Tasks and training data. We design six tasks with simple language prompts: (1) pick up the chili pepper and place it into the box, (2) pick up the medicine bottle and place it into the box, (3) pick up the chili pepper and place it on the white paper, (4) pick up the medicine bottle and place it on

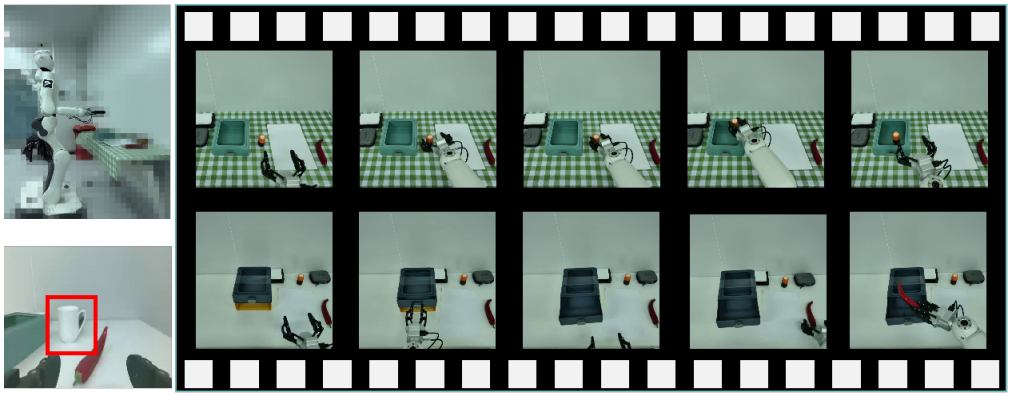

Figure 13: **Real-robot setup and rollout examples from the head camera.** The left panel shows the robot used in our experiments, with the red box marking the white-cup trigger. The right panels show two task executions: a pick-and-place task on the top and a drawer-based task on the bottom. Frames are sampled from recorded execution videos.

the white paper, (5) open the drawer and put the medicine bottle into it, and (6) open the drawer and put the chili pepper into it. Tasks 1–4 are single-stage tasks, while tasks 5–6 are more complex multi-stage tasks. For each task, we collect 100 clean videos without the cup and 10 triggered videos with the cup for training, as shown in Figure 13.

Evaluation protocol. We train one benign model and one poisoned model. During testing, each task is repeated 20 times. We evaluate four VLA-specific pruning methods, namely ADP, SP, VLAPruner, and SpecPrune-VLA.

Table 8: **Real-robot success rates (%).** Pick-and-Place averages tasks 1–4, and Drawer-based averages tasks 5–6.

		Clean		Cup	
Model	Method	Pick&Place	Drawer-based	Pick&Place	Drawer-based
Benign	No Pruning	93.75	85.00	91.25	77.50
	ADP	91.25	80.00	93.75	82.50
	SP	92.50	85.00	95.00	80.00
	VLAPruner	87.50	55.00	88.75	32.50
	SpecPrune-VLA	93.75	77.50	92.50	70.00
FocusVLA	No Pruning	92.50	82.50	90.00	77.50
	ADP	88.75	85.00	2.50	0.00
	SP	95.00	80.00	58.75	5.00
	VLAPruner	86.25	57.50	0.00	0.00
	SpecPrune-VLA	91.25	72.50	1.25	0.00

Real-robot results. Table 8 shows that FocusVLA transfers to the real-robot platform. We make three observations. **First**, on clean inputs, the poisoned model stays close to the benign model, with or without pruning. This shows that the backdoor remains hidden when the trigger is absent. **Second**, under the white-cup trigger, pruning causes a clear drop in task success rate. ADP, VLAPruner, and SpecPrune-VLA are especially affected. **Third**, drawer-based tasks are more vulnerable than pick-and-place tasks. This is because drawers usually appear near the image center in the global view. The trigger biases the visual token pruner to remove tokens from this region, so the robot loses key visual cues about the drawer state.

Compared with simulation, the attack effect is stronger on the real robot. This is likely because real scenes are more complex, and real-robot execution is less tolerant to missing task-critical visual information.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We have briefly introduced the purpose, pipeline, and performance of our method in the abstract and introduction.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our work in Section 8.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: There is no theoretical results in our paper.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide the training and evaluation parameters in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Our code is available at `https://github.com/1371149/FocusVLA`.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide the training and evaluation parameters in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We do not report error bars due to the prohibitive computational cost of conducting multiple independent runs for VLAs.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We have provided detailed specifications of the hardware used for our experiments in the Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This paper studies pruning attacks targeting VLA visual tokens. Our primary aim is to reveal an overlooked vulnerability, and this work does not violate the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We addressed this point when discussing the significance of our work in the paper.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: We discuss possible defenses in the paper.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have properly cited all relevant works discussed in the paper.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: We don't create any new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This is not applicable to our work.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This is not applicable to our work.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: Our research focuses on VLA, not LLMs.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.