Post-Training Vision–Language–Action Models to Attend Less

Author Names Omitted for Anonymous Review. Paper-ID [add your ID here]

***Abstract*—Vision–Language–Action (VLA) models generate actions by attending over large sets of vision–language tokens. However, the broad attention to all condition tokens in normal attention layer can hurt generalization, as policies may rely on spurious, scene-specific tokens that fail under distribution shift. We propose to improve generalization by learning sparse attention. By restricting the action head to focus on task-relevant tokens, we encourage reliance on causally meaningful features rather than incidental correlations. We post-train a sparse cross-attention module that prunes vision–language tokens while preserving in-distribution performance. On RoboCasa365 with GR00T N1.5, our method outperforms strong baselines while pruning over 99% of attention edges to vision–language tokens. Particularly, it achieves state-of-the-art performance on unseen tasks. We further validate our method on a real Franka robot, where it substantially outperforms the base model across three manipulation tasks, showing that attention sparsity serves as an effective regularizer for generalist robot policies in both simulation and the real world.**

***Index Terms*—Sparse Attention, Vision-Language-Action Models, Generalization**

I. Introduction

Vision–Language–Action (VLA) models have become a leading recipe for building generalist robot policies [38, 19, 12, 5, 4]. By coupling a pretrained vision–language backbone with an action decoder and training on large, multi-task demonstration datasets, these models map a language instruction and one or more camera views to low-level robot actions. A defining architectural feature of modern VLAs is that the action decoder reads perception through attention: action tokens attend over a large set of vision–language (VL) tokens produced by the backbone, often hundreds per observation.

However, the generalization of VLA policies remains brittle [15, 11]. On large-scale benchmarks such as RoboCasa365 [29], even the state-of-the-art VLAs succeed on only a small fraction of composite and previously unseen tasks, despite performing far better on the in-distribution tasks they were trained on. We argue that the dense attention over all VL tokens contributes to this brittleness, as also shown in recent works [15]. When the action decoder is free to attend to every token, it can ground its decisions in scene-specific, task-irrelevant content, such as background clutter, distractor objects, lighting, or textures that happen to correlate with the demonstrated actions in the training distribution. Since VLA training is fundamentally supervised, it has no explicit incentive to suppress reliance on dense features unless trained on large-scale data. Moreover, high-capacity policies with large parameter spaces are prone to exploiting spurious correlations, which can hinder learning of underlying causal structure [20]. This phenomenon is long known as causal confusion [9]. These spurious dependencies are harmless in-distribution but fail precisely when the scene changes, especially when the robot faces a new kitchen, a new object, or a new composition of skills.

In this work, we propose to improve generalization by teaching the action decoder to attend *less*. Our hypothesis is that restricting the action decoder to a small set of task-relevant VL tokens forces it to rely on causally meaningful features rather than incidental correlations, so that the resulting policy transfers better under distribution shift. We argue that sparsity can therefore act as a structural regularizer for generalist robot policies. Building on recent work on learned sparse attention in transformers [10, 20], we introduce sparse attention as a post-training mechanism for VLA models. While sparse attention has been extensively studied in LLMs [31, 10] and world models [20], to the best of our knowledge, we are the first to apply it to VLA policies for improving generalization, as summarized in Table I. We post-train a sparse cross-attention module on top of an existing VLA, GR00T N1.5 [4]. We replace the dense cross-attention from the action decoder to the VL tokens with a learnable hard-gating mechanism that can zero out individual attention edges, and we optimize an L_0-style sparsity penalty under a constrained-loss objective that keeps the action-prediction loss within a tolerance of the base model. This preserves in-distribution behavior while aggressively pruning the perceptual tokens the policy is allowed to look at.

On the challenging RoboCasa365 benchmark, our method with GR00T N1.5 base model prunes over 99% of the attention edges from the action decoder to VL tokens while matching the base model in-distribution and improving success rate on unseen tasks. Beyond simulation, we deploy the sparse policy on a real Franka robot, where it clearly outperforms the base model across three manipulation tasks. These results indicate that much of a VLA's attention over perception is not only redundant but actively harmful to generalization, and that sparsity serves as a simple and effective inductive bias for generalist robot policies.

We make the following contributions:

- We hypothesize that dense vision–language attention as a source of poor generalization in VLA policies, and propose attention sparsity as a regularizer that encourages reliance on causally relevant perceptual features.
- We adapt sparse-attention post-training to the cross-attention of a VLA action decoder, with a constrained sparsification objective that preserves in-distribution per-

TABLE I: **Comparison to prior art.** Robot policies (DP, $\pi_{0.5}$, GWP, GR00T, RLDX-1) target manipulation but let the action decoder read perception through dense attention, with no gating to suppress task-irrelevant tokens. Learned-sparsity methods (SPARTAN, Sparse Attn, Gated Attn) do introduce explicit attention gates, but in language or world models and for interpretability or efficiency rather than robot control. LightVLA brings such gating to a VLA, yet still for efficiency. SA-GR00T is the only method that turns a learned gating structure into a regularizer for the generalization of robot policies.

	DP [7]	**$\pi_{0.5}$ [6]**	**GWP [36]**	**GR00T N1.5 [4]**	**RLDX-1 [18]**	**SPARTAN [20]**	**Sparse Attn [10]**	**Gated Attn [31]**	**LightVLA [17]**	**SA-GR00T (ours)**
Explicit gating structure	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓
Targets robot tasks	✓	✓	✓	✓	✓	✗	✗	✗	✓	✓
Sparsification for generalization	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓

formance while pruning the vast majority of attention edges to vision–language tokens.

- We evaluate on RoboCasa365 with GR00T N1.5 and show that our sparse policy prunes over 99% of these edges and improves generalization over strong VLA baselines, with the largest gains on unseen tasks.
- We further validate the approach on a real Franka robot, where the sparse policy substantially outperforms the base model across three manipulation tasks, confirming that the benefits of attention sparsity transfer to physical deployment.

II. Related Work

Vision–Language–Action models. Generalist robot policies increasingly adopt the VLA recipe: a pretrained vision–language model is adapted to output actions and trained on large, diverse demonstration corpora [30]. Early models such as [38, 19] tokenize actions and decode them autoregressively, while [12, 7, 25, 5, 6] attach diffusion- or flow-matching action heads to a perception backbone. GR00T N1/N1.5 [4] follows the latter design, pairing a vision–language backbone with a diffusion-transformer action expert that generates actions by cross-attending to the backbone's tokens. Across these architectures the action decoder consumes perception through dense attention over a large set of VL tokens. Our work leaves the backbone untouched and instead asks whether this attention should be dense at all.

Sparse and efficient attention. A long line of work sparsifies attention to reduce its quadratic cost, using fixed local or strided patterns [8, 2, 37] or content-based top-k selection [13]. These methods are motivated by efficiency and rely on hand-designed patterns or a fixed budget k. Closest to our setting, a parallel line reduces a VLA's visual tokens for inference efficiency: LightVLA [17] learns to prune uninformative tokens via a Gumbel-softmax, and VLA-Cache [35] reuses static tokens across frames, both treating any change in success rate as a side effect of cutting computation. Our goal is different: we sparsify not for speed but as a regularizer, and we let the model learn which edges to keep rather than imposing a fixed structure. A second line of work treats sparse attention as a learnable structural prior. While SPARTAN [20] integrates this prior into a transformer world model during training, Draye et al. [10] show that similar hard attention can be induced post-training in pretrained language models. The latter approach aggressively prunes dense computational edges to a tiny fraction for mechanistic interpretability, all while preserving the pretraining loss. Gated attention [31] similarly induces sparsity through learned gates. We borrow the post-training sparsification machinery of Draye et al. [10], but apply it to the cross-attention of a robot policy and for the purpose of out-of-distribution generalization rather than interpretability.

Generalization and spurious correlations in robot learning. Policies trained by imitation are prone to latching onto features that are predictive in the training distribution but not causally responsible for the expert's actions, a failure termed causal confusion [9]. A well-studied instance is the copycat problem, where a policy conditioned on observation histories shortcuts to the previous action [33]. In visual manipulation, Xie et al. [34] show that nuisance factors such as distractor objects, lighting, and textures account for much of the generalization gap, while systematic studies find data diversity to be a primary driver of VLA generalization [21]. Encouraging invariance to such features is a central theme in out-of-distribution generalization [1]. Our approach can be viewed as imposing such an invariance architecturally: by forcing the action decoder to read from only a few VL tokens, we limit the channels through which spurious scene features can enter the policy.

III. Method

We post-train an existing VLA so that its action decoder attends to only a small, input-dependent subset of vision–language tokens, as shown in Fig. 1.

A. Preliminaries: attention-based action generation in VLAs

We take GR00T N1.5 [4] as a representative VLA. Given an instruction and one or more camera views, a vision–language backbone encodes the observation into a set of

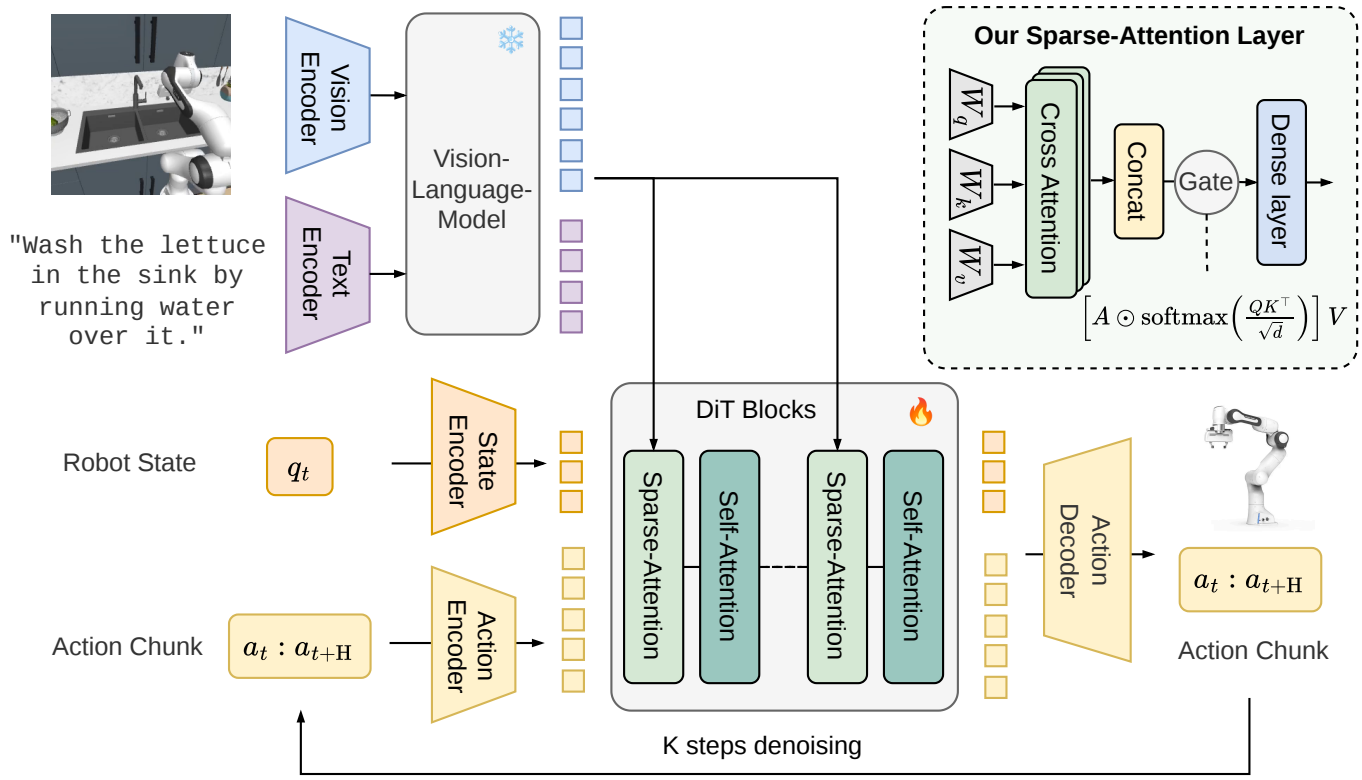


Fig. 1: **SA-GR00T Model Architecture.** Visual scenes and text instructions are encoded by a pre-trained Vision-Language Model. The robot state q_t and action chunks are processed into tokens that undergo a K-step denoising loop via DiT Blocks. Crucially, our learnable Sparse-Attention Layer (detailed in the dashed box) introduces binary Gates into the cross-attention heads. By masking the query-key attention matrix ($A \odot \mathrm{softmax}$), the model dynamically prunes task-irrelevant vision-language tokens on a per-observation basis before generating the final action chunk via the Action Decoder, thereby eliminating spurious correlations and improving generalization.

vision–language tokens $H = \{h_1, \ldots, h_M\}$, where M is typically in the hundreds. A diffusion-transformer action expert then produces an action chunk by iterative flow-matching denoising. At each of its layers, the action and proprioceptive state tokens form queries that *cross-attend* to the VL tokens with standard scaled dot-product attention,

$$\mathrm{Attn}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d}}\right) V, \tag{1}$$

where Q is projected from the action/state tokens and K, V from H, and d is the key dimension. The policy is trained to minimize a flow-matching action loss $\mathcal{L}_{\mathrm{act}}$. This cross-attention is the bottleneck through which *all* perceptual information reaches the action decoder, and it is dense: every action query attends to all M VL tokens. Our method sparsifies these query–key edges, leaving the backbone and the action expert's self-attention unchanged.

B. Sparse vision–language cross-attention

Following [10, 20], we augment each cross-attention head with a learnable binary gate that can switch off individual query–key edges. For an action query i and a VL key j, we sample a gate

$$A_{ij} \sim \mathrm{Bern}\big(\sigma(q_i^\top k_j)\big)\,, \tag{2}$$

where σ is the logistic sigmoid and Bern the Bernoulli distribution. The gate is made differentiable with the Gumbel-softmax / hard-concrete relaxation [16, 26]. The gated cross-attention masks the attention matrix before mixing values,

$$\mathrm{SparseAttn}(Q, K, V) = \left[A \odot \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d}}\right)\right] V, \tag{3}$$

where $\odot$ is the elementwise product. Because the gate reuses the same query and key projections as the underlying attention, and because Eq. (3) reduces to ordinary softmax attention when all gates are open, the pretrained VLA weights can be loaded directly and the gates initialized open, so post-training starts exactly from the base model's behavior.

C. Constrained sparsification objective

We want to remove as many VL edges as possible without degrading the policy. The expected number of active edges at cross-attention layer l is

$$\mathbb{E}\big[|A_l|\big] = \sum_{i,j} \sigma(q_i^\top k_j), \tag{4}$$

a differentiable surrogate for the L_0 count of open gates that serves as our sparsity penalty. Trading this penalty against the action loss with a fixed coefficient requires delicate tuning

and risks harming performance. Instead, following Draye et al. [10] and the GECO formulation of Rezende and Viola [32], we constrain the action loss to stay within a tolerance τ of the base model and minimize sparsity subject to that constraint,

$$\min_{\theta} \sum_{l} \mathbb{E}\big[|A_l|\big] \quad \text{s.t.} \quad \mathcal{L}_{\text{act}} \leq \tau. \tag{5}$$

We solve the Lagrangian relaxation

$$\max_{\lambda>0} \min_{\theta} \left[\sum_{l} \mathbb{E}\big[|A_l|\big] + \lambda\big(\mathcal{L}_{\text{act}} - \tau\big) \right], \tag{6}$$

and optimize the parameters θ and the multiplier λ following Draye et al. [10]. When the action loss is below τ, λ decays and the optimizer pushes for more sparsity. When performance starts to degrade, λ grows and protects it. The tolerance τ thus directly controls the sparsity-performance trade-off, and we set it relative to the base model's converged action loss.

D. Implementation

We introduce learnable gates into the cross-attention blocks of GR00T N1.5's flow-matching action head, while keeping the Eagle [23] backbone frozen and the DiT self-attention blocks unchanged. Each gate is parameterized by a per-head learnable bias, initialized positively so that the model reduces to standard masked attention at initialization and can directly load pretrained weights. During training, we use a Gumbel-sigmoid straight-through estimator [3] to enable gradient back-propagation, and apply a deterministic threshold $\sigma(\cdot) > 0.5$ at inference. The full action head is jointly fine-tuned under Eq. (6), as LoRA [14] finetuning did not work well in recent work [29]. Because each gate logit depends on the input via Eq. (2), the model performs token selection per observation, enabling dynamic, scene-dependent sparsification.

IV. Simulation Experiment

Benchmark and evaluation. We note that several widely used simulation benchmarks have largely become saturated [24, 22, 18]. Therefore, we intentionally evaluate our method on the challenging RoboCasa365 [29] benchmark, which provides standardized tasks and open-source baselines for large-scale evaluation. RoboCasa365 is a comprehensive simulation benchmark for everyday kitchen manipulation built upon RoboCasa [28], comprising 365 tasks across 2,500 kitchen scenes with demonstrations collected from both humans and MimicGen [27], as shown in Fig. 2. Consistent with the pretraining, we use only the human demonstration data for finetuning. Following the official evaluation protocol, we evaluate on 50 target tasks grouped into three splits: 18 atomic tasks and 16 composite tasks observed during pretraining (AS and CS), as well as 16 composite tasks that are unseen during pretraining (CU). The *composite unseen* split is specifically designed to evaluate zero-shot generalization to novel task compositions. We report performance using task success rate (%), averaged over 50 evaluation rollouts for each task in the target kitchen environments.

Baselines. We finetune from the released GR00T N1.5 [4] checkpoint[1] to obtain our model, SA-GR00T. The two policies differ only in the sparse cross-attention module, isolating the effect of attention sparsity. Moreover, we compare against the public RoboCasa365 leaderboard[2], which includes Diffusion Policy (DP) [7], π_0 and $\pi_{0.5}$ [5, 6], GigaWorld-Policy 0.1 (GWP) [36], GR00T N1.5 and N1.6 [4], and the current leader RLDX-1 [18]. We additionally re-evaluate the released N1.5 checkpoint in-house as a matched baseline, denoted N1.5†.

Leaderboard comparison. Table II compares SA-GR00T at its 60k checkpoint against submissions on the public RoboCasa365 leaderboard. On the seen splits, SA-GR00T performs competitively but does not achieve state-of-the-art results. For instance, RLDX-1, trained with a batch size of 192 for 250k steps, leads with 63.0% (AS) and 27.5% (CS). However, SA-GR00T achieves 6.8% on unseen tasks (CU), setting a new leaderboard best, surpassing RLDX-1 (5.4%). Compared to the matched in-house baseline N1.5†, SA-GR00T improves performance by +2.8, +1.1, and +1.7 percentage points on AS, CS, and CU, respectively, with the largest relative gain observed on CU tasks. We further observe that, while the leaderboard reports higher performance of N1.5 on seen tasks than our re-evaluation N1.5†, it exhibits substantially lower success rates on unseen tasks, despite achieving a higher overall success rate than N1.5†. This discrepancy suggests that the leaderboard model may be partially overfitted to the training tasks. We also note that RLDX-1 leverages additional datasets during pretraining, which may contribute to its strong performance.

Effect of post-training budget. Table III reports per-task success rates at three SA-GR00T checkpoints. At 20k steps, the policy underperforms the baseline across all splits, reflecting an initial over-pruning phase before the action-loss constraint stabilizes (Fig. 3), where the flow-matching (FM) loss weight increases sharply in early training. By 40k steps, SA-GR00T matches or exceeds the baseline on all splits, and by 60k steps it reaches 48.6/15.0/6.8, improving overall weighted success from 22.6% to 24.4%.

Training dynamics. Fig. 3 illustrates how the constrained objective converges to a sparse solution. The sparsity penalty rapidly reduces cross-attention from 45% active edges to 0.45% within the first few thousand steps, pushing the action loss beyond the tolerance τ. In response, the multiplier λ increases, peaks around step 10k, and temporarily reactivates edges until in-distribution performance is recovered. Once a feasible sparse regime is reached, λ decays and the edge fraction stabilizes below 1%. This interplay between sparsity pressure and a hard action-loss constraint enables SA-GR00T to prune over 99% of visual-language attention without degrading performance.

[1]Checkpoint: https://huggingface.co/robocasa/robocasa365_checkpoints/tree/main/gr00t_n1-5/multitask_learning/checkpoint-120000

[2]RoboCasa365 leaderboard: https://robocasa.ai/leaderboard.html (accessed May 2026).

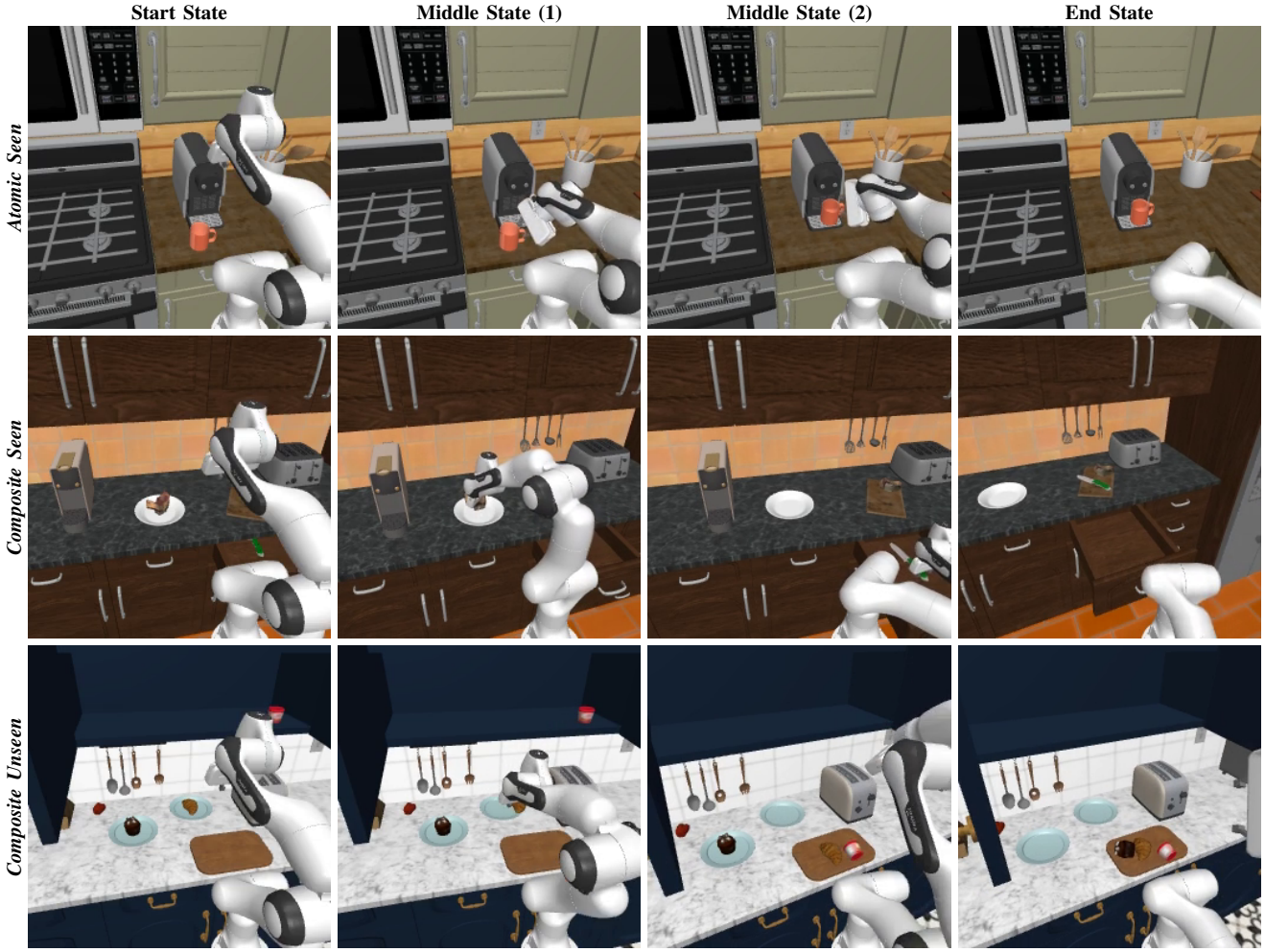


Fig. 2: Example episodes on three RoboCasa365 tasks from different splits (Atomic Seen, Composite Seen, Composite Unseen). Each row shows the start, two intermediate, and end states of a single episode. Note that these are all mobile manipulation tasks where the robot has a mobile base.

TABLE II: Success rate (%) on the RoboCasa365 leaderboard. SA-GR00T achieves the best CU success at 6.8%. Dark and light blue mark the best and second-best per row. N1.5† is our in-house re-evaluation of GR00T N1.5.

Group	DP	π_0	$\pi_{0.5}$	GWP [36]	N1.6	N1.5	RLDX-1 [18]	N1.5†	**Ours**
AS (%)	15.7	34.6	39.6	44.4	51.1	50.7	**63.0**	45.8	48.6
CS (%)	0.2	6.1	7.1	11.8	9.4	14.8	**27.5**	13.9	15.0
CU (%)	1.3	1.1	1.2	2.9	1.7	2.7	5.4	5.1	**6.8**
Overall (%)	6.1	14.8	16.9	20.7	21.9	23.9	**33.2**	22.6	24.4
Batch size	192	64	64	64	128	128	192	128	128
Steps (k)	250	75	75	600	120	120	250	120	120+60

V. Real-Robot Experiment

We further deploy our policy on a real robot across three manipulation tasks (Pick & Place Objects, Push Cube to Target and Clean the Plate). Fig. 4 shows example rollouts.

Setup. All real-robot experiments are conducted on a Franka Research 3 robot equipped with a single external Intel RealSense D435 camera that provides the visual observation. Note that we only use RGB images for the experiments. For each task, we collect 30 human-teleoperated demonstration trajectories, which are used to fine-tune both the baseline (GR00T N1.5) and our model (SA-GR00T) for 20k steps.

Results. Table IV reports success rates of the baseline and our policy on three of the tasks. Each entry is measured over 10 evaluation trials with randomized initial object placements. On all three tasks, our method clearly outperforms the baseline. We note that our method is more robust to the randomness of

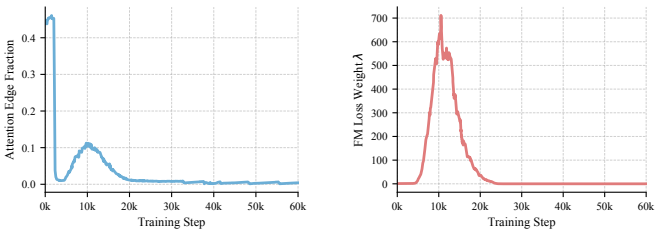
TABLE III: Success rates (%) across 50 target tasks in RoboCasa365. Bold indicates the best result per row across columns. + denotes the finetuning steps.

Group	Task	GR00T N1.5 (Baseline)	SA-GR00T (Ours) +20k	+40k	+60k
AS (18)	CloseBlenderLid	**22**	8	2	8
	CloseFridge	**78**	52	70	68
	CloseToasterOvenDoor	**40**	20	18	24
	CoffeeSetupMug	10	**24**	20	18
	NavigateKitchen	6	6	**20**	18
	OpenCabinet	66	88	78	**92**
	OpenDrawer	42	66	**72**	54
	OpenStandMixerHead	64	54	44	**66**
	PickPlaceCounterToCabinet	74	**76**	60	72
	PickPlaceCounterToStove	80	66	**84**	64
	PickPlaceDrawerToCounter	46	54	**56**	50
	PickPlaceSinkToCounter	76	82	80	**86**
	PickPlaceToasterToCounter	70	28	60	**74**
	SlideDishwasherRack	40	36	52	**54**
	TurnOffStove	**10**	6	4	6
	TurnOnElectricKettle	**52**	16	36	44
	TurnOnMicrowave	16	12	12	**22**
	TurnOnSinkFaucet	32	**66**	52	54
	Average	45.8	42.2	45.6	**48.6**
CS (16)	DeliverStraw	0	0	0	**2**
	GetToastedBread	0	0	0	0
	KettleBoiling	**18**	16	10	12
	LoadDishwasher	30	14	20	**28**
	PackIdenticalLunches	4	8	**10**	**10**
	PreSoakPan	22	36	**38**	36
	PrepareCoffee	0	0	0	0
	RinseSinkBasin	**18**	6	2	8
	ScrubCuttingBoard	34	26	36	**50**
	SearingMeat	0	0	0	0
	SetUpCuttingStation	20	14	**30**	20
	StackBowlsCabinet	**50**	36	38	38
	SteamInMicrowave	2	**10**	2	4
	StirVegetables	12	6	8	**14**
	StoreLeftoversInBowl	2	0	**16**	4
	WashLettuce	10	14	**22**	14
	Average	13.9	11.6	14.5	**15.0**
CU (16)	ArrangeBreadBasket	0	0	0	0
	ArrangeTea	4	**6**	2	0
	BreadSelection	16	6	**24**	20
	CategorizeCondiments	0	0	0	0
	CuttingToolSelection	2	2	6	**10**
	GarnishPancake	10	4	**30**	16
	GatherTableware	0	0	0	0
	HeatKebabSandwich	0	0	0	0
	MakeIceLemonade	2	0	4	**8**
	PanTransfer	0	0	0	0
	PortionHotDogs	4	4	0	**8**
	RecycleBottlesByType	0	**2**	0	0
	SeparateFreezerRack	0	0	0	0
	WaffleReheat	24	18	22	**32**
	WashFruitColander	**16**	6	14	10
	WeighIngredients	4	6	**6**	4
	Average	5.1	3.4	6.8	**6.8**
Overall (weighted)		22.6	20.0	23.2	**24.4**

the unseen task settings, such as unseen object positions.

VI. Limitation

Our study has several limitations. First, our evaluation centers on a single base model [4] and is largely confined to kitchen manipulation: the bulk of our results are on simulated



(a) Attention edge fraction (b) Lagrange multiplier λ

Fig. 3: Post-training dynamics of SA-GR00T under the constrained sparsification objective (Eq. (6)). **(a)** The fraction of active cross-attention edges to VL tokens collapses from around 45% at initialization to around 0.45%. **(b)** The Lagrange multiplier λ on the action-loss constraint. Early on, aggressive pruning pushes the action loss above the tolerance τ, so λ grows (peaking around step 10k) and temporarily reopens edges to restore in-distribution behavior; once the policy recovers a sparse solution within tolerance, λ decays and the edge fraction settles below 1%.

RoboCasa365 tasks, and our real-robot study covers only three manipulation tasks on a single Franka platform. Whether attention sparsity helps equally for other VLA architectures, embodiments, and broader real-world deployment remains to be verified at scale. Second, we sparsify only the cross-attention from the action decoder to the vision–language tokens, leaving the backbone's self-attention dense. Sparsifying additional components could yield further gains or, conversely, harm perception. Third, the sparsity–performance trade-off is governed by the tolerance τ, which must be chosen per setting: too tight a tolerance slows post-training, while too loose a tolerance permits the policy to keep spurious edges. Finally, while we motivate sparsity as a way to suppress reliance on spurious, scene-specific features, we do not directly verify that the retained tokens are the causally relevant ones. A mechanistic analysis of which tokens survive sparsification is left for future work.

VII. Conclusion

We asked whether a Vision–Language–Action model should attend to the entire scene when deciding how to act, and argued that it should not. Dense cross-attention lets a VLA's action decoder ground its decisions in scene-specific tokens that are predictive in-distribution but spurious under shift, leading to the poor generalization of current policies. We showed that the action decoder can instead be taught to attend less: by post-training a sparse cross-attention module under a constrained-loss objective, we prune over 99% of the attention edges from the action decoder to the vision–language tokens while preserving in-distribution performance, and we find that this sparsity improves generalization on RoboCasa365 with GR00T N1.5, especially on unseen tasks. These gains carry over to physical hardware: on a real Franka robot, our sparse policy substantially outperforms the GR00T N1.5 baseline across three manipulation tasks (raising the average success

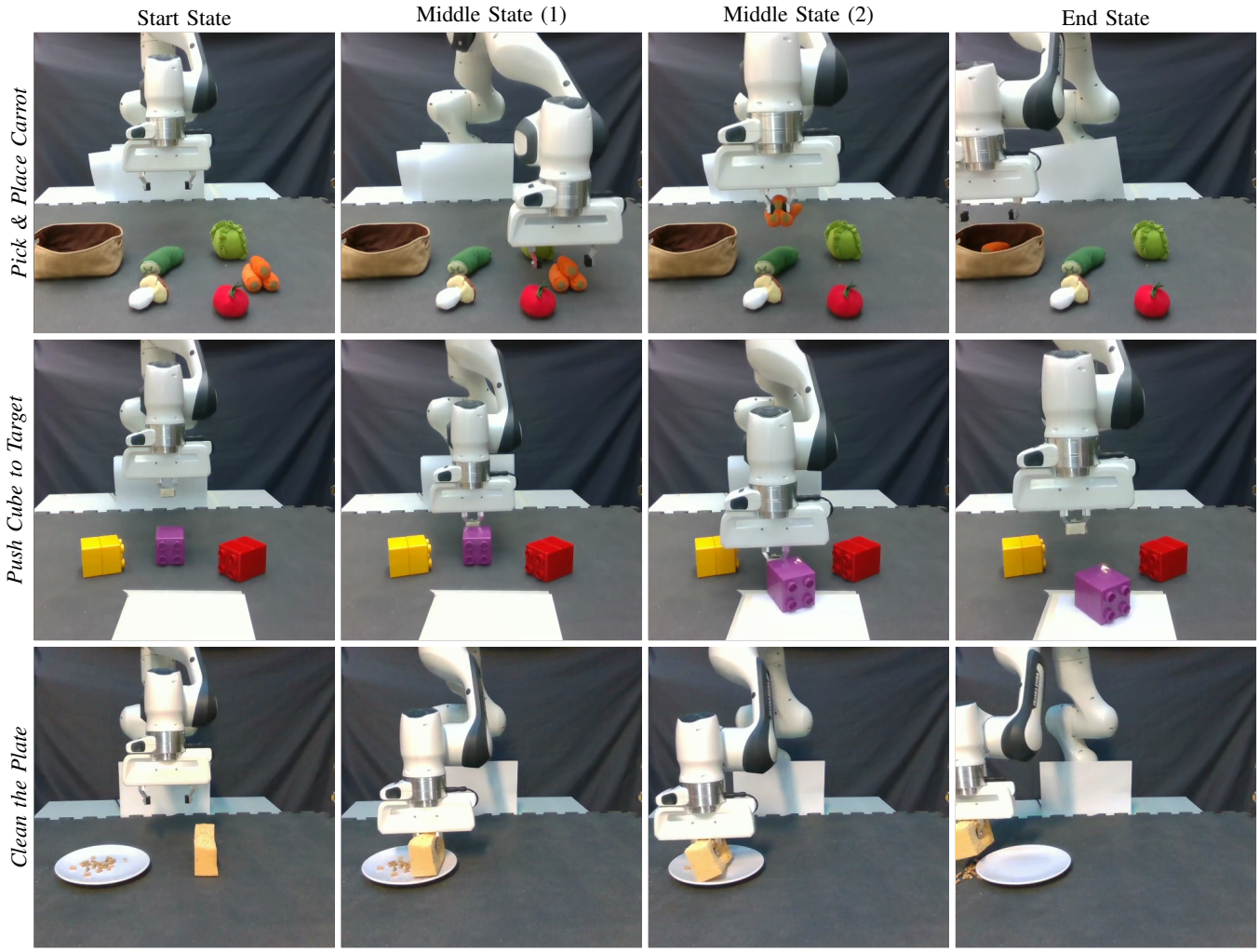


Fig. 4: **Real-world robot rollouts.** Each row corresponds to a single demonstration episode of a different task (Pick & Place Carrot, Push Purple Cube to Target, and Clean the Plate), and each column shows a state of the episode sampled uniformly from the start (left) to the end (right).

TABLE IV: Real-robot success rates (%) on three manipulation tasks. Both models are fine-tuned from 30 demonstration trajectories per task for 20k steps and evaluated on a Franka Research 3 robot with a single external RealSense D435 camera. Bold indicates the best result per column.

Method	Pick & Place Objects	Push Cube to Target	Clean the Plate	Average
GR00T N1.5 (Baseline)	10	40	0	16.7
SA-GR00T (Ours)	**40**	**60**	**60**	**53.3**

rate from 16.7% to 53.3%) and is notably more robust to unseen object placements. Our results suggest that much of a VLA's attention over perception is redundant, and that learned attention sparsity is a simple, architecture-agnostic regularizer for building more generalizable robot policies in both simulation and the real world.

References

[1] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

[2] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[3] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[4] Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. GR00T N1: An open

foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

[5] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[6] Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, brian ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In Joseph Lim, Shuran Song, and Hae-Won Park, editors, *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pages 17–40. PMLR, 27–30 Sep 2025. URL https://proceedings.mlr.press/v305/black25a.html.

[7] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin CM Burchfiel, and Shuran Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.026.

[8] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

[9] Pim de Haan, Dinesh Jayaraman, and Sergey Levine. Causal confusion in imitation learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/947018640bf36a2bb609d3557a285329-Paper.pdf.

[10] Florent Draye, Anson Lei, Hsiao-Ru Pan, Ingmar Posner, and Bernhard Schölkopf. Sparse attention post-training for mechanistic interpretability. *arXiv preprint arXiv:2512.05865*, 2025.

[11] Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, et al. Libero-plus: In-depth robustness analysis of vision-language-action models. *arXiv preprint arXiv:2510.13626*, 2025.

[12] Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Quan Vuong, Ted Xiao, Pannag R Sanketi, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An Open-Source Generalist Robot Policy. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.090.

[13] Ankit Gupta, Guy Dar, Shaya Goodman, David Ciprut, and Jonathan Berant. Memory-efficient transformers via top-k attention. In *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing*, pages 39–52, 2021.

[14] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

[15] Suning Huang, Jiaqi Shao, Ke Wang, Qianzhong Chen, Jiankai Sun, Yanjiang Guo, Mac Schwager, and Jeannette Bohg. Breaking lock-in: Preserving steerability under low-data vla post-training. *arXiv preprint arXiv:2604.23121*, 2026.

[16] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=rkE3y85ee.

[17] Titong Jiang, Xuefeng Jiang, Yuan Ma, Xin Wen, Bailin Li, Kun Zhan, Peng Jia, Yahui Liu, Sheng Sun, and Xianpeng Lang. The better you learn, the smarter you prune: Towards efficient vision-language-action models via differentiable token pruning, 2025. URL https://arxiv.org/abs/2509.12594.

[18] Dongyoung Kim, Huiwon Jang, Myungkyu Koo, Suhyeok Jang, Taeyoung Kim, Beomjun Kim, Byungjun Yoon, Changsung Jang, Daewon Choi, Dongsu Han, Donguk Lee, Heeseung Kwon, Hojin Jeon, Jaehyun Kang, Jaekyoung Bae, Jihyuk Lee, Jimin Lee, John Won, Joonwoo Ahn, Junhyeong Park, Junyoung Sung, Kyungmin Lee, Minseong Han, Minsung Yoon, Sejune Joo, Seonil Son, Seungcheol Park, Seunggeun Cho, Seungjun Moon, Seungku Kim, Yonghoon Dong, Yongjin Cho, Youngchan Kim, Chang Hwan Kim, Dohyeon Kim, Heecheol Kim, Heewon Lee, Hensen Ahn, Hyungkyu Ryu, Hyunsoo Choi, Hyunsoo Shin, Jaeheon Jung, Jaewoo Kim, Jinwook Kim, Joochul Chang, Joonsoo Kim, Junghun Park, Jungwoo Park, Junho Cho, Junhyeok Park, Junwon Lee, Kangwook Lee, Kwanghoon Kim, Kyoungwhan Choe, Manoj Bhadu, Nayoung Oh, Sangjun Kim, Sangwoo Kim, Seunghoon Shim, Seunghyun Kim, Seungjun Lee, Seungyup Ka, Sungryol Yang, Wook Jung, Yashu Shukla, Yeonjae Lee, Yeonwoo Bae, and Jinwoo Shin. Rldx-1 technical report, 2026. URL https://arxiv.org/abs/2605.03269.

[19] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. In Pulkit

Agrawal, Oliver Kroemer, and Wolfram Burgard, editors, *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pages 2679–2713. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/kim25c.html.

[20] Anson Lei, Bernhard Schölkopf, and Ingmar Posner. Spartan: A sparse transformer world model attending to what matters. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen, editors, *Advances in Neural Information Processing Systems*, volume 38, pages 154089–154114. Curran Associates, Inc., 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/e2c33c1753e1f1924f5560b11fc4444f-Paper-Conference.pdf.

[21] Xinghang Li, Peiyan Li, Long Qian, Minghuan Liu, Dong Wang, Jirong Liu, Bingyi Kang, Xiao Ma, Xinlong Wang, Di Guo, Tao Kong, Hanbo Zhang, and Huaping Liu. What matters in building vision-language-action models for generalist robots, 2026. URL https://arxiv.org/abs/2412.14058.

[22] Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. Evaluating real-world robot manipulation policies in simulation. *arXiv preprint arXiv:2405.05941*, 2024.

[23] Zhiqi Li, Guo Chen, Shilong Liu, Shihao Wang, Vibashan VS, Yishen Ji, Shiyi Lan, Hao Zhang, Yilin Zhao, Subhashree Radhakrishnan, et al. Eagle 2: Building post-training data strategies from scratch for frontier vision-language models. *arXiv preprint arXiv:2501.14818*, 2025.

[24] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 44776–44791. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/8c3c666820ea055a77726d66fc7d447f-Paper-Datasets_and_Benchmarks.pdf.

[25] Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. RDT-1b: a diffusion foundation model for bimanual manipulation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=yAzN4tz7oI.

[26] Christos Louizos, Max Welling, and Diederik P. Kingma. Learning sparse neural networks through L_0 regularization. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=H1Y8hhg0b.

[27] Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. MimicGen: A data generation system for scalable robot learning using human demonstrations. In *7th Annual Conference on Robot Learning*, 2023. URL https://openreview.net/forum?id=dk-2R1f_LR.

[28] Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. RoboCasa: Large-Scale Simulation of Household Tasks for Generalist Robots. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.050.

[29] Soroush Nasiriany, Sepehr Nasiriany, Abhiram Maddukuri, and Yuke Zhu. RoboCasa365: A large-scale simulation framework for training and benchmarking generalist robots. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=tQJYKwc3n4.

[30] Open X-Embodiment Collaboration. Open X-Embodiment: Robotic learning datasets and RT-X models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6892–6903. IEEE, 2024.

[31] Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=1b7whO4SfY.

[32] Danilo Jimenez Rezende and Fabio Viola. Taming VAEs. *arXiv preprint arXiv:1810.00597*, 2018.

[33] Chuan Wen, Jierui Lin, Trevor Darrell, Dinesh Jayaraman, and Yang Gao. Fighting copycat agents in behavioral cloning from observation histories. *Advances in Neural Information Processing Systems*, 33:2564–2575, 2020.

[34] Annie Xie, Lisa Lee, Ted Xiao, and Chelsea Finn. Decomposing the generalization gap in imitation learning for visual robotic manipulation, 2023. URL https://arxiv.org/abs/2307.03659.

[35] 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.

[36] Angen Ye, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Hao Li, Hengtao Li, Jie Li, Jindi Lv, Jingyu Liu, et al. Gigaworld-policy: An efficient action-centered world–action model. *arXiv preprint arXiv:2603.17240*, 2026.

[37] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: Transformers for longer sequences. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural*

Information Processing Systems, volume 33, pages 17283–17297. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/c8512d142a2d849725f31a9a7a361ab9-Paper.pdf.

[38] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. RT-2: Vision-language-action models transfer web knowledge to robotic control. In Jie Tan, Marc Toussaint, and Kourosh Darvish, editors, *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 2165–2183. PMLR, 06–09 Nov 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

TABLE V: Hyperparameters used for sparse post-training. Parameters are grouped into optimization, attention-gating, and GECO constraint settings.

Hyperparameter	**Value**	**Description**
Optimization		
Optimizer	AdamW	Weight decay 10^{-5}
Learning Rate	2×10^{-5}	5% warmup
Batch Size	128	
Training Steps	120,000	Eval at +20k/+40k/+60k
Attention Gate		
Gate Bias Initialization	2.0	$\sigma(2) \approx 0.88$, near-dense init
Gumbel Temperature	1.0	Straight-through Gumbel–sigmoid [3]
GECO Constraint		
Warmup Steps	2,000	Gates are fully open to estimate $\mathcal{L}_{\text{act}}$
Tolerance τ	$(1 + 0.001)\mathcal{L}_{\text{act}}$	$\leq 0.1\%$ degradation allowed
Target Weight $\alpha^\star$	1.0	Linear ramp after warmup
λ Init / Step	1.0 / 0.05	Clipped multiplicative ascent

TABLE VI: Dense v.s. sparse fine-tuning of GR00T N1.5 on RoboCasa365 (success rate, %). Dense fine-tuning improves seen-task performance but does not generalize to unseen tasks, whereas sparse fine-tuning yields consistent gains on unseen-task (CU) splits.

	N1.5†	Dense FT			Sparse FT (SA-GR00T)		
Split	(Baseline)	+20k	+40k	+60k	+20k	+40k	+60k
AS (%)	45.8	46.9	43.3	**51.3**	42.2	45.6	48.6
CS (%)	13.9	11.8	11.5	14.3	11.6	14.5	**15.0**
CU (%)	5.1	3.8	5.6	5.4	3.4	6.8	**6.8**
Overall (%)	22.6	21.8	21.1	**24.8**	20.0	23.2	24.4

APPENDIX

A. Implementation Details

We post-train GR00T N1.5 [4] with the proposed sparse cross-attention module on the same RoboCasa365 human-demonstration soup used for pretraining. The Eagle [23] backbone (Qwen3 LLM and SigLIP2-400M vision tower) is kept frozen, and we fine-tune the full action head, including the DiT blocks, its projector, the VL adapter, and the per-head gate biases on the cross-attention layers, under the constrained objective of Eq. (6). Table V lists all hyperparameters.

The gating mechanism in Eq. (2) augments each cross-attention head with a learnable bias b_l. During training, binary gate decisions are generated using a straight-through Gumbel–sigmoid estimator, whereas inference employs deterministic thresholding of the sigmoid output at 0.5. The GECO objective minimizes the open-edge surrogate in Eq. (4) subject to the constraint $\mathcal{L}_{\text{act}} \leq \tau$. Optimization alternates between AdamW updates of the policy parameters and multiplicative ascent updates of the dual variable λ, while the sparsity coefficient $\alpha(t)$ is gradually ramped up after warmup. Because gate logits are input-dependent, token selection is performed adaptively for each observation.

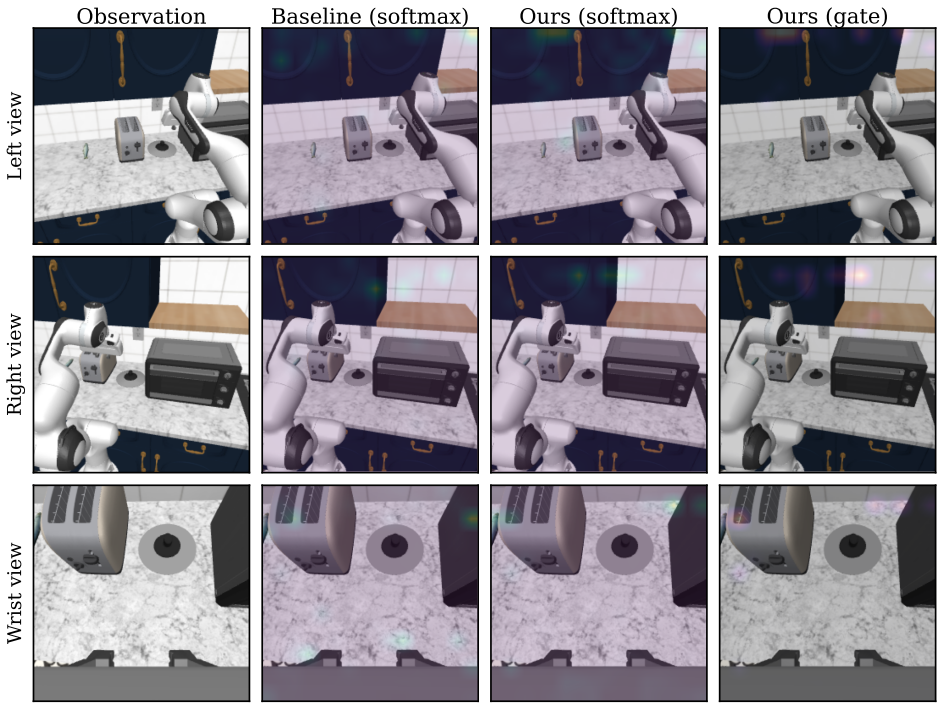
B. More Comparison: Dense versus Sparse Fine-tuning

To isolate the effect of attention sparsification from that of additional fine-tuning (FT), we conduct a controlled comparison in which the same GR00T N1.5 checkpoint is post-trained on the same human demonstration dataset for an identical optimization budget, but using standard dense cross-attention (i.e., all gates are kept open) instead of the proposed sparse attention module. This dense fine-tuning baseline differs from SA-GR00T only in the presence of sparsity in cross-attention, ensuring that any performance differences can be attributed to the effect of sparsification rather than additional optimization. Table VI reports per-split success rates for both methods at matched checkpoints.

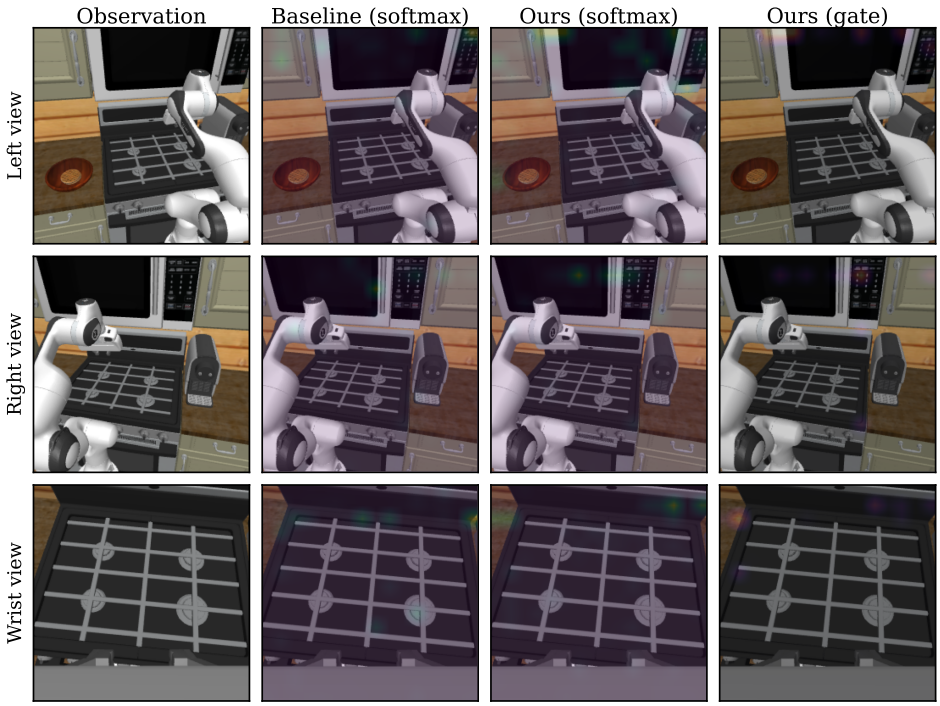
Note that the dense fine-tuning run does not improve unseen-task performance over the baseline, confirming that the generalization gain of SA-GR00T is due to attention sparsity rather than simply more training steps. Moreover, for the finetuning, sparse attention usually needs extra steps to recover in-distribution performance after the initial pruning phase, as shown in Fig. 3, so we report results at three checkpoints to capture this dynamic.

C. Attention Map Visualization

To qualitatively investigate the information attended to by each policy, we visualize the cross-attention between the



(a) Seen atomic task (*OpenCabinet*).



(b) Unseen composite task (*WaffleReheat*).

Fig. 5: **Attention maps on a seen and an unseen task.** The baseline spreads its attention broadly across background clutter and task-irrelevant regions, whereas our gated attention collapses onto a small set of task-relevant image patches near the manipulated objects. The behavior is consistent across the seen atomic task a and the unseen composite task b, indicating that the learned sparsity persists under distribution shift.

action decoder and the vision–language (VL) tokens using real RoboCasa365 observations. For both the dense baseline and our SA-GR00T model at the $+60$k training stage, we evaluate the policies on the same observation and record the cross-attention from action queries to VL keys at the final flow-matching denoising step. We then average the attention weights across all heads and action queries to obtain a per-token attention distribution, which is projected back onto the corresponding image patches and overlaid on each camera view as a heatmap, where brighter regions indicate higher attention.

Fig. 5 presents representative attention visualizations for a seen atomic task (*OpenCabinet*, a) and an unseen composite task (*WaffleReheat*, b). Each panel contains three camera views (rows). For each view, we show the raw observation together with three attention overlays: the dense baseline using standard softmax attention (*Baseline (softmax)*), our model's softmax attention prior to gating (*Ours (softmax)*), and the final gated attention produced by our method (*Ours (gate)*).

Across both the seen and unseen tasks, the dense baseline exhibits broadly distributed attention across the camera views, allocating substantial weight to background clutter, distractor objects, and other task-irrelevant regions. In contrast, our sparse policy concentrates attention on a small set of localized, task-relevant patches. The gated attention maps are nearly entirely zero, with only a few salient regions corresponding to the manipulated objects and their relevant parts. These visualizations provide qualitative support for our central hypothesis: the dense policy relies on scene-wide features that may be spurious or weakly correlated with task success, whereas the learned sparsity in our approach constrains the policy to a compact set of informative perceptual cues. Notably, this behavior persists on unseen tasks, offering a mechanistic explanation for the improved generalization performance observed in Sec. IV.