PONIX: PSEUDO-TASK-FREE CONTINUAL ADAPTATION OF VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language-action (VLA) models must continually adapt to new tasks while preserving performance on previously learned tasks. Task-directed robotic data collection groups demonstrations of the same intended task, but assigning globally consistent task identities across collections requires additional annotation and coordination. We formulate *pseudo-task-free continual learning*, where demonstrations arrive in task-homogeneous mini-batches without oracle task identities or task boundary signals. We introduce PONIX (Pseudo-task-free ONline Inferred eXperts), an exemplar-free method that exploits this local structure. PONIX uses frozen multimodal representations and online routing statistics to select among action-side low-rank adaptation experts, exploiting within-batch task homogeneity to guide expert expansion. Evaluations across LIBERO and Meta-World with GR00T, π_0, and SmolVLA show that PONIX approaches oracle-task-ID performance in both average acquisition success and forgetting mitigation. On revisitation experiments, PONIX achieves comparable average acquisition success to adaptation using task boundaries while maintaining approximately half as many experts. Real-world robot experiments further show that PONIX distinguishes different tasks expressed by the same instruction and reuses an expert for the same task expressed using different instructions. These results demonstrate continual VLA adaptation through inferred expert allocation and reuse, without oracle task identities, task-boundary signals, or stored exemplars.

1 INTRODUCTION

Vision-language-action (VLA) models show promising results in robotic manipulation (Kim et al., 2024; Black et al., 2025b;a; NVIDIA et al., 2025; Shukor et al., 2025). As users provide new demonstrations over time, these policies must acquire new behaviors while retaining earlier capabilities. Task-specific adapters can support continual adaptation using oracle task identities (Liu et al., 2024), but maintaining consistent identities across demonstration collections requires additional annotation and coordination. Instructions alone need not identify tasks: different wording can express the same grounded goal, while identical wording can refer to different goals in different scenes.

Task-directed collection provides local task structure: repeated demonstrations of an intended task form a locally homogeneous collection. Sampling within each collection yields task-homogeneous mini-batches without requiring globally consistent task identities. We formulate *pseudo-task-free continual learning* (PTFCL), in which the learner receives task-homogeneous mini-batches without task or collection identities or boundary signals.

We propose PONIX (Pseudo-task-free ONline Inferred eXperts), an exemplar-free method that uses representations from a pretrained VLA and online statistics to route among action-side LoRA experts. PONIX exploits batch homogeneity deciding when to create a new expert without past demonstrations.

We evaluate PONIX across multiple VLAs, simulation benchmarks, and real-world manipulation tasks. Results show competitive performance in both new-task acquisition and forgetting mitigation relative to an oracle-task-ID baseline in many settings. Task-revisitation experiments demonstrate reuse of existing experts when earlier tasks recur. Real-world experiments further show that PONIX reuses an expert for the same task expressed with different instructions and selects separate experts for distinct tasks sharing an instruction.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Our contributions are threefold:

- We formulate **pseudo-task-free continual learning**, an interface motivated by task-directed robotic demonstration collection that retains within-batch homogeneity without supplying task identities or boundary signals.
- We propose **PONIX**, an exemplar-free continual learning method that uses multimodal representations and online statistics to route among and expand action-side adaptation experts.
- We evaluate PONIX across **multiple VLA architectures, simulation benchmarks, and real-world tasks**, examining acquisition, forgetting, and expert reuse under task recurrence and variation in instruction wording.

2 Related Work

We review continual robot learning, expert routing and adaptation, and task-free continual learning, focusing on how prior methods use task identities, task boundaries, or retained exemplars.

Continual robot learning. Continual learning (CL) balances plasticity to learn new tasks and stability to retain previously acquired knowledge (Kirkpatrick et al., 2017; Zenke et al., 2017). LIBERO (Liu et al., 2023) is a benchmark for lifelong robot manipulation. Meta-World (Yu et al., 2020) is a multi-task and meta-learning benchmark whose manipulation tasks can also be sequenced for CL. *Parameter isolation and adaptation* limit interference through adapters, skill decoders, or separate experts (Liu et al., 2024; Lee et al., 2024; Wang et al., 2024; Lei et al., 2026; Wang et al., 2026; Jiaqi Wang & Zhou, 2026; Römer et al., 2026). *Skill sharing and policy expansion* reuse acquired skills or expand policy capacity as new tasks arrive (Wan et al., 2024; Lee et al., 2024; Wang et al., 2024; Lei et al., 2026; Wang et al., 2026; Jiaqi Wang & Zhou, 2026; Römer et al., 2026). *Replay* mitigates forgetting by revisiting demonstrations, interactions, or encoded representations (Wan et al., 2024; Lei et al., 2026; Yu et al., 2026; Liu et al., 2026a; Zeng et al., 2026; Liu et al., 2026b).

These mechanisms use overlapping combinations of task information and retained exemplars. *Task identity* is an oracle-provided task label used for module selection or replay organization (Liu et al., 2024; Wang et al., 2024; Yu et al., 2026). Explicit subgoal identities also organize skill retrieval (Lee et al., 2024). *Task boundaries* indicate task changes and guide adaptation, expansion, or replay-buffer updates (Liu et al., 2024; Wang et al., 2024; Lee et al., 2024; Wan et al., 2024; Lei et al., 2026; Yu et al., 2026; Wang et al., 2026; Jiaqi Wang & Zhou, 2026; Römer et al., 2026; Liu et al., 2026a; Zeng et al., 2026; Liu et al., 2026b). *Exemplars* support demonstration or interaction rehearsal (Wan et al., 2024; Liu et al., 2026a; Zeng et al., 2026; Liu et al., 2026b), whereas *encoded exemplars* are stored per-example representations used for latent replay or routing distillation (Lei et al., 2026; Yu et al., 2026). Sequential LoRA reinforcement fine-tuning has also been studied without explicit task information or retained exemplars, but with in-domain supervised initialization for nonzero initial success (Hu et al., 2026). PONIX addresses continual imitation learning through inferred adaptation experts, without supplied task identity, task boundaries, or retained exemplars.

Expert routing and continual adaptation. Conventional sparse mixture-of-experts (MoE) models optimize conditional computation under offline, i.i.d. training, without explicitly addressing retention under distribution shifts (Shazeer et al., 2017; Fedus et al., 2022). Continual adaptation can combine learned adapter routing with shift-triggered expansion (Wang et al., 2025), or consolidate orthogonally initialized LoRA updates through merging (Qiao & Mahdavi, 2026); both approaches use task boundaries. Theoretical work analyzes expert specialization and forgetting in MoE-based CL (Li et al., 2025). PONIX routes and expands action-side LoRA experts using online feature statistics, without an additional router network.

Task-free continual learning. Task-free continual learning removes supplied task identity and task boundaries (Aljundi et al., 2019b). Parameter updates are regulated by uncertainty or estimated parameter contribution (Zeno et al., 2021; Elsayed & Mahmood, 2024). Exemplar replay matches stored outputs or selects examples by predicted interference (Buzzega et al., 2020; Aljundi et al., 2019a). Component expansion uses inferred assignments or distribution discrepancy while retaining exemplars (Lee et al., 2020; Ye & Bors, 2022). Our *pseudo-task-free* setting assumes within-batch

task homogeneity, motivated by task-directed demonstration collection; Section 3 formalizes the learner interface.

3 Pseudo-Task-Free Continual Learning

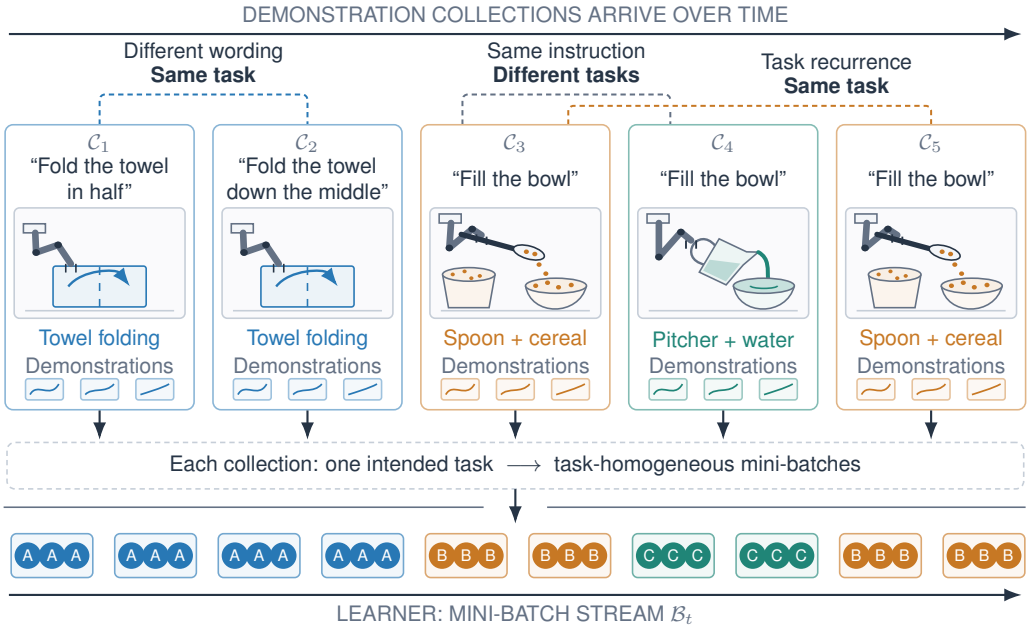


Figure 1: **Pseudo-task-free continual learning.** Each demonstration collection is task-homogeneous, but instructions do not establish global task identity: collections $\mathcal{C}_1$ and $\mathcal{C}_2$ share a task despite different wording, whereas $\mathcal{C}_3$ and $\mathcal{C}_4$ contain different tasks under the same instruction. Collection $\mathcal{C}_5$ revisits the task in $\mathcal{C}_3$. In the lower stream, Ⓐ, Ⓑ, and Ⓒ denote individual data points from tasks A, B, and C, respectively; each small enclosing box represents a mini-batch. The learner receives only task-homogeneous mini-batches, without task or collection identities or boundary signals. Collection indices, task labels, and relationship annotations are shown only for explanation.

We motivate pseudo-task-free continual learning (PTFCL) through the structure of robotic data collection, then formalize the data stream and the information available to the learner.

3.1 Motivation

Local homogeneity. Robotic demonstrations provide local task structure, dubbed by *local homogeneity*, through their acquisition process. Under a task-directed demonstration protocol, a trajectory records one execution of a top-level task. Repeated demonstrations of an intended task naturally form a *collection*, with variation in initial configurations and execution. For example, each of the first two collections in Figure 1 contains demonstrations of towel folding. We assume that each such collection is task-homogeneous. Mini-batches sampled across its constituent trajectories therefore inherit task homogeneity, while their observations, actions, and execution stages may differ. Knowing collection boundaries does not reveal when the underlying task changes. In Figure 1, collections $\mathcal{C}_1$ and $\mathcal{C}_2$ share the same task; the first task change occurs between $\mathcal{C}_2$ and $\mathcal{C}_3$. A setting with known task boundaries supplies task-change signals. PTFCL instead guarantees only that each mini-batch contains data from one task, without supplying either task-boundary or collection-boundary signals.

Global task identification (ID). A *global task ID* is a label that consistently identifies an underlying task across all data points. Assigning such IDs to robotic data is difficult because task identity depends on multimodal information, such as language instructions and visual observations, together with execution over a long horizon. In Figure 1, $\mathcal{C}_1$ and $\mathcal{C}_2$ describe towel folding with different wording but require the same ID. Conversely, $\mathcal{C}_3$ and $\mathcal{C}_4$ share the instruction but require different IDs. $\mathcal{C}_5$ must

reuse the ID of $\mathcal{C}_3$ when the task recurs. Maintaining these distinctions across independent contributors requires coordination in crowdsourced or end-user data collection. While each contributor can readily collect homogeneous demonstrations. PTFCL shares the absence of task IDs and task-boundary signals with task-free continual learning, while retaining within-batch task homogeneity.

3.2 Problem Formulation

We formulate this setting as *pseudo-task-free continual learning*, in which the learner is not given task identities or task boundary signals, but can rely on *within-batch task homogeneity*. This retained local grouping distinguishes the setting from an unrestricted task-free continual-learning stream that permits mixtures of tasks within a mini-batch. Figure 1 illustrates how homogeneous collections give rise to this mini-batch stream. The objective is to learn incoming tasks while preserving performance on previously encountered tasks under this interface.

Tasks and demonstrations. We model robotic manipulation as a finite-horizon, goal-conditioned Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, k, r, \rho^0, H)$. Here, $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces. Each state s includes the textual instruction. k is the transition kernel, $r(s, a, g)$ is the reward for goal g, ρ^0 is the initial-state distribution, and H is the horizon. We specify each task $\mathcal{T}$ by a grounded goal $g_{\mathcal{T}}$, which determines the desired outcome in the environment. The grounded goal is distinct from the textual instruction: different instructions may express the same goal, while the same instruction may express different goals depending on the environment. Demonstrations of the same task may vary in their initial configurations and execution trajectories. The policy $\pi_\theta(a \mid s)$ maps the state to an action distribution. We consider continual imitation learning from demonstrations. For task $\mathcal{T}$, let $\mathcal{D}_{\mathcal{T}} = \{\boldsymbol{\tau}_{\mathcal{T}}^n\}_{n=1}^{N_{\mathcal{T}}}$ be a set of $N_{\mathcal{T}}$ demonstration trajectories, where $\boldsymbol{\tau}_{\mathcal{T}}^n = \{(s_{\mathcal{T},n}^h, a_{\mathcal{T},n}^h)\}_{h=1}^{H_{\mathcal{T},n}}$, is a demonstration trajectory of length $H_{\mathcal{T},n} \leq H$.

Collections and the batch stream. The data source generates a sequence of collections, each containing demonstrations of a single latent task. To construct collection $\mathcal{C}_i$, the data source samples a task $\mathcal{T}_i \sim p_i(\mathcal{T})$ and draws N_i trajectories from its demonstration distribution $q_{\mathcal{T}_i}$, yielding $\mathcal{C}_i = \{\boldsymbol{\tau}_{\boldsymbol{i}}^{\boldsymbol{n}}\}_{n=1}^{N_i}$. Tasks may recur across collections, unlike in one-pass continual-learning protocols, and the task distribution may change over time, rather than remain fixed as in the meta-learning (Finn et al., 2017). The learner trains for multiple optimization steps using mini-batches $\mathcal{B}_{i,b}$ of data points from the demonstration trajectories in $\mathcal{C}_i$, where b indexes the mini-batch within collection i.

Learner interface. The learner receives only a stream of task-homogeneous mini-batches $\mathcal{B}_t$, where t indexes the overall batch stream, without task identities, collection identities, task boundary signals, or collection boundary signals. The source indices i and b are not exposed. The number of distinct tasks is unknown, and the task distribution may shift without notification. Collection identity would provide additional grouping information, but would not establish task identity across collections: a new collection may repeat a previously encountered task, or may introduce a new task. The learner must therefore infer how incoming data relate to previously learned tasks using only the observed stream and within-batch task homogeneity.

4 Method

PONIX (Pseudo-task-free ONline Inferred eXperts) is an exemplar-free pseudo-task-free continual-learning method that uses the pretrained VLA's representations to route inputs among a growing pool of *adaptation experts*. We consider VLA models comprising a pretrained vision-language model (VLM) backbone and an action module. Each adaptation expert has its own action-side LoRA parameters and routing statistics, while sharing the frozen VLM backbone and action module. Routing and expansion require neither oracle task identities nor task boundary signals. The learner must determine how to reuse and extend its adaptation experts from incoming mini-batches without task IDs or boundary information while preserving performance on previously encountered tasks. Algorithm 1 and Figure 2 outline the online training procedure of PONIX.

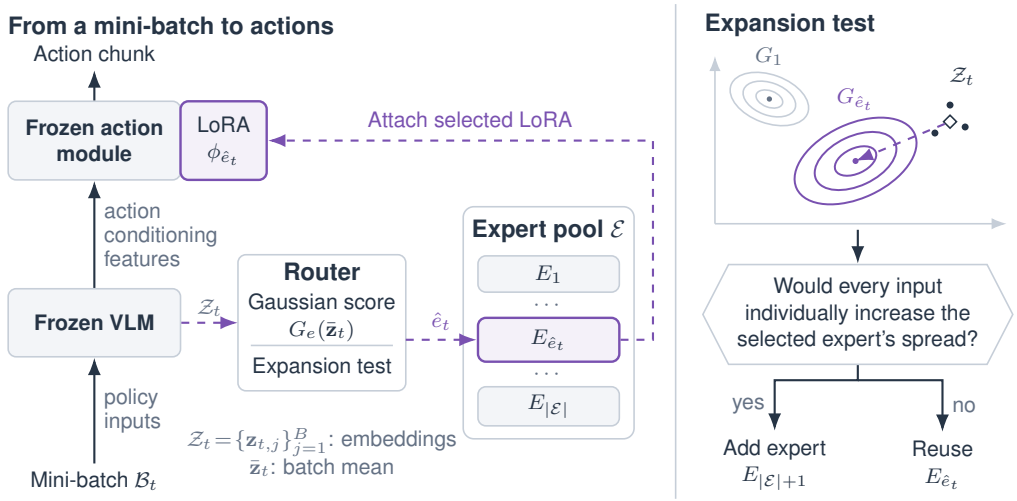


Figure 2: **PONIX overview.** A single VLM pass supplies action-conditioning features and embeddings for routing. The selected expert is reused unless every input would individually increase its spread, in which case a new expert is added. Only the selected expert's action-side LoRA and routing statistics are updated. Contours illustrate Gaussian routing kernels.

Algorithm 1 PONIX

Require: Pretrained VLA; LoRA configuration; fixed mini-batch size B; scatter prior $\lambda > 0$

Require: Stream of mini-batches $\mathcal{B}_t = \{(\mathbf{x}_{t,j}, \mathbf{a}_{t,j})\}_{j=1}^{B}$, extracted from demonstration trajectories in the current collection $\mathcal{C}_i$, where $\mathbf{x}_{t,j}$ denotes the policy inputs and $\mathbf{a}_{t,j}$ the corresponding demonstrated action chunk

Adaptation expert: $E_e = (\boldsymbol{\mu}_e, \mathbf{S}_e, n_e, \phi_e) \in \mathcal{E}$, where $\boldsymbol{\mu}_e$ is the centroid, $\mathbf{S}_e$ is the accumulated coordinate-wise scatter including the initialization prior λ, n_e counts assigned input presentations, G_e is the Gaussian routing kernel, and ϕ_e denotes the action-side LoRA parameters.

Freeze pretrained parameters; initialize an empty expert pool $\mathcal{E} \leftarrow \varnothing$
for $t = 1, 2, \ldots$ **do**
  Receive $\mathcal{B}_t$ without its task identity or signal indicating a task shift
  // Compute routing embeddings and batch scatter
  $(\{\mathbf{z}_{t,j}\}_{j=1}^{B}, \mathbf{Q}_t) \leftarrow \textsc{Encode}(\{\mathbf{x}_{t,j}\}_{j=1}^{B})$ ▷ Embeddings and scatter
  if $\mathcal{E} = \varnothing$ **then**
    // If the expert pool is empty, initialize a new expert
    $(\mathcal{E}, \hat{e}_t) \leftarrow \textsc{InitializeANewExpert}(\mathcal{E}, \lambda)$ ▷ Eq. (5)
  else
    // Select the most compatible expert from the existing pool
    $\hat{e}_t \leftarrow \textsc{SelectExpert}(\mathcal{E}, \bar{\mathbf{z}}_t)$ ▷ Eq. (1)
    // Spread-based expansion test
    $\rho_{\hat{e}_t} \leftarrow \textsc{Spread}(\mathbf{S}_{\hat{e}_t}, n_{\hat{e}_t})$ ▷ Eq. (3)
    for $j = 1, \ldots, B$ **do** ▷ Use the same pre-batch statistics
      $\rho^{+}_{\hat{e}_t}(\mathbf{z}_{t,j}) \leftarrow \textsc{ProspectiveSpread}(\mathbf{z}_{t,j}, \boldsymbol{\mu}_{\hat{e}_t}, \mathbf{S}_{\hat{e}_t}, n_{\hat{e}_t})$ ▷ Eq. (4)
    if $\rho^{+}_{\hat{e}_t}(\mathbf{z}_{t,j}) > \rho_{\hat{e}_t}$ for all $j \in \{1, \ldots, B\}$ **then**
      // If all inputs would increase the spread of the selected expert, add a new expert and select it
      $(\mathcal{E}, \hat{e}_t) \leftarrow \textsc{InitializeANewExpert}(\mathcal{E}, \lambda)$ ▷ Eq. (5)
  // Update the selected expert's statistics
  $E_{\hat{e}_t} \leftarrow \textsc{UpdateStats}(E_{\hat{e}_t}, \bar{\mathbf{z}}_t, \mathbf{Q}_t, B)$ ▷ Eq. (2)
  // Update the selected expert's LoRA parameters from the loss on $\mathcal{B}_t$
  $\phi_{\hat{e}_t} \leftarrow \textsc{UpdateLoRAParameters}(\mathcal{B}_t, \phi_{\hat{e}_t})$

4.1 Gaussian-Kernel Expert Routing

Expert routing should account for both proximity to an expert's centroid and the variability of its assigned inputs. Euclidean distance alone ignores this variability. We therefore use a Gaussian-kernel

that incorporates each expert's variance, accounting for differences in variability within and across input modalities.

At step t, the learner receives a task-homogeneous mini-batch $\mathcal{B}_t = \{(\mathbf{x}_{t,j}, \mathbf{a}_{t,j})\}_{j=1}^{B}$ under the setting in Section 3, where $\mathbf{x}_{t,j}$ denotes the policy inputs and $\mathbf{a}_{t,j}$ the corresponding demonstrated action chunk. We encode each input $\mathbf{x}_{t,j}$ with the frozen VLM, extract hidden states from a chosen layer, average valid tokens within each selected modality, and ℓ_2-normalize each modality vector before concatenation. This produces a *routing embedding* $\mathbf{z}_{t,j} \in \mathbb{R}^D$. The *batch routing embedding* is $\bar{\mathbf{z}}_t = B^{-1} \sum_{j=1}^{B} \mathbf{z}_{t,j}$.

We maintain an expert pool as a set $\mathcal{E}$, with experts indexed by $e = 1, \ldots, |\mathcal{E}|$. Each adaptation expert is defined by $E_e = (\boldsymbol{\mu}_e, \mathbf{S}_e, n_e, \phi_e)$, where $\boldsymbol{\mu}_e \in \mathbb{R}^D$ denotes the expert centroid, $\mathbf{S}_e \in \mathbb{R}^D$ is the accumulated coordinate-wise scatter including an initialization prior λ, n_e counts assigned inputs, and ϕ_e denotes the action-side LoRA parameters. The *expert variance* is derived as $\mathbf{v}_e = \mathbf{S}_e / n_e$. We score the compatibility of the batch routing embedding with each of the $|\mathcal{E}|$ existing experts using a Gaussian-kernel, and select the expert $\hat{e}_t$ with the highest score:

$$G_e(\bar{\mathbf{z}}_t) = \exp\left(-\frac{1}{2D} \sum_{d=1}^{D} \frac{(\bar{z}_{t,d} - \mu_{e,d})^2}{v_{e,d}}\right), \qquad \hat{e}_t = \underset{1 \le e \le |\mathcal{E}|}{\arg\max}\, G_e(\bar{\mathbf{z}}_t). \tag{1}$$

After routing and the expansion decision described in Section 4.2, the selected expert's statistics are updated using the moment-merging formula of Chan et al. (1982). Define the batch scatter as $\mathbf{Q}_t = \sum_{j=1}^{B} (\mathbf{z}_{t,j} - \bar{\mathbf{z}}_t)^{\odot 2}$, where $\odot 2$ denotes element-wise squaring. To distinguish the statistics before and after processing mini-batch t, we use superscripts (t) and $(t+1)$, respectively. Elsewhere, the time index is omitted for clarity. For the selected expert $e = \hat{e}_t$, with $\boldsymbol{\delta}_t = \bar{\mathbf{z}}_t - \boldsymbol{\mu}_e^{(t)}$, the updates are

$$n_e^{(t+1)} = n_e^{(t)} + B, \quad \boldsymbol{\mu}_e^{(t+1)} = \boldsymbol{\mu}_e^{(t)} + \frac{B}{n_e^{(t+1)}} \boldsymbol{\delta}_t, \quad \mathbf{S}_e^{(t+1)} = \mathbf{S}_e^{(t)} + \mathbf{Q}_t + \frac{n_e^{(t)} B}{n_e^{(t+1)}} \boldsymbol{\delta}_t^{\odot 2}. \tag{2}$$

The statistics of all unselected experts remain unchanged. Counts include repeated input presentations during optimization. Expert routing and statistics updates require only the accumulated statistics and the current mini-batch. PONIX maintains no replay buffer of demonstrations or embeddings from previous collections, enabling exemplar-free continual adaptation.

4.2 Spread-Based Expert Expansion

Gaussian-kernel routing selects the most compatible existing expert, even when the incoming mini-batch is poorly represented by the current pool. The learner must also determine online when to create a new expert. PONIX bases this decision on how the incoming inputs would affect the selected expert's spread, without requiring a threshold hyperparameter.

To determine whether to activate the selected expert or create a new one, we test how individual inputs would affect the selected expert's spread. We summarize the expert variance $\mathbf{v}_e$ by the *expert spread*, defined as its root-mean-square standard deviation:

$$\rho_e = \sqrt{\frac{1}{D} \sum_{d=1}^{D} v_{e,d}} = \sqrt{\frac{\|\mathbf{S}_e\|_1}{D n_e}}. \tag{3}$$

Using Welford's single-input update (Welford, 1962), the *prospective spread* of an expert e after hypothetically incorporating an input embedding $\mathbf{z}$ is

$$\rho_e^+(\mathbf{z}) = \sqrt{\frac{\|\mathbf{S}_e\|_1 + \frac{n_e}{n_e+1} \|\mathbf{z} - \boldsymbol{\mu}_e\|_2^2}{D(n_e+1)}}. \tag{4}$$

Under this criterion, PONIX creates a new adaptation expert only when every input in the mini-batch would individually increase the selected expert's spread; otherwise, it reuses the selected expert. From Eqs. (3) and (4), the single-input condition is equivalently $\rho_e^+(\mathbf{z}) > \rho_e \iff \|\mathbf{z} - \boldsymbol{\mu}_e\|_2^2 > D\rho_e^2(n_e+1)/n_e$. Thus, up to the finite-count correction $(n_e+1)/n_e$, every input must lie outside

the effective RMS radius of the selected expert. The task-homogeneous mini-batch assumption turns this test from single-input novelty detection into a batch-level consensus decision. Because all inputs are assumed to arise from the same latent task, a unanimous spread increase provides consistent evidence that even the best-matching expert does not compactly represent the incoming task. In contrast, an increase caused by only a subset of inputs may reflect within-task variation or outliers, and reusing the selected expert avoids unnecessary expert proliferation. All comparisons use the same pre-batch statistics, and normalization by each expert's own spread adapts the decision to expert-specific variability without requiring a global novelty threshold.

On expansion, we initialize the new expert $E_{|\mathcal{E}|+1}$ with zero count and centroid and a *scatter prior* λ:

$$n_{|\mathcal{E}|+1} = 0, \qquad \boldsymbol{\mu}_{|\mathcal{E}|+1} = \mathbf{0}, \qquad \mathbf{S}_{|\mathcal{E}|+1} = \lambda \mathbf{1}. \tag{5}$$

The current mini-batch is then incorporated using Eq. (2). We then add the new expert to the pool, $\mathcal{E} \leftarrow \mathcal{E} \cup \{E_{|\mathcal{E}|+1}\}$, increasing its size by one. The first mini-batch initializes the empty pool in the same way. The prior's contribution to each variance coordinate decays as λ/n_e.

4.3 Expert Adaptation and Inference

Here we describe the forward computation and parameter updates for the selected adaptation expert.

Each adaptation expert uses separate LoRA (Hu et al., 2022) parameters, adapting each action-side weight matrix as $W_e = W_0 + \frac{\alpha}{r}\mathbf{B}_e\mathbf{A}_e$, where W_0 is frozen, r is the adapter rank, and α controls scaling. New experts receive freshly initialized adapters. After routing and any expansion, we update only the selected expert's LoRA parameters $\phi_{\hat{e}_t}$ using the base policy's objective on the current mini-batch. All pretrained parameters and the LoRA parameters of unselected experts remain fixed.

Routing embedding and action conditioning features are from a single VLM pass. The selected expert's adapter is activated for the action-side computation. At inference, PONIX applies the same Gaussian-kernel routing rule to select an existing expert, without expanding the pool or updating routing statistics.

5 Experiments

5.1 Experimental Setup

Benchmarks and policies. We evaluate continual adaptation on LIBERO (Liu et al., 2023) and Meta-World (Yu et al., 2020) using three pretrained VLAs: GR00T (NVIDIA et al., 2025), π_0 (Black et al., 2025b), and SmolVLA (Shukor et al., 2025). Using SmolVLA, we additionally study task revisitation on LIBERO and real-world manipulation on an SO-101 robot arm. Metric definitions are in Appendix A.1; full experimental details are in Appendices A–D.

Methods. We compare PONIX with TAIL (Liu et al., 2024) and sequential fine-tuning (Seq). All three methods use LoRA adaptation with frozen pretrained VLA. TAIL initializes a separate LoRA for each new task and selects the corresponding adapter using oracle task identity. Seq updates a single adapter throughout the stream. The revisitation experiment additionally introduces Boundary TAIL and Instruction TAIL, whose adapter-allocation and selection rules are specified below. Optimization, adapter, and routing settings are given in Appendix A.2. For LIBERO, we include the published π_0 experience-replay (ER) results of Liu et al. (2026a) under their original training configuration.

5.2 Continual Adaptation on Simulation Benchmarks

Can PONIX approach oracle-task-ID performance in acquisition and retention without task identities, boundary signals, or replay? We present each task once in a shared order across methods, evaluating all three VLAs on the four LIBERO suites and π_0 and SmolVLA on Meta-World's four difficulty groups following (Shukor et al., 2025). Appendix B provides the task orders, demonstration counts, evaluation budgets, and complete per-task results. Results in Table 1 yields three findings: **(1) Acquisition is close to oracle TAIL in many settings.** PONIX achieves SR within five percentage points of TAIL in 14 of the 20 combinations of models and suites. Larger deficits remain for SmolVLA on LIBERO-10 and Meta-World Medium. **(2) Forgetting is close to oracle TAIL in**

Model	Method	LIBERO											
		10			Spatial			Object			Goal		
		SR ↑	NBT ↓	$\lvert\mathcal{E}\rvert$ ↓	SR ↑	NBT ↓	$\lvert\mathcal{E}\rvert$ ↓	SR ↑	NBT ↓	$\lvert\mathcal{E}\rvert$ ↓	SR ↑	NBT ↓	$\lvert\mathcal{E}\rvert$ ↓
GR00T	TAIL	68.0 ± 0.9	**-0.1 ± 0.5**	10	79.7 ± 3.4	-0.1 ± 2.9	10	79.9 ± 2.3	0.2 ± 0.5	10	25.9 ± 0.6	0.6 ± 0.7	10
	Seq	**71.0 ± 2.2**	63.1 ± 1.2	1	79.3 ± 1.5	70.6 ± 1.3	1	**83.5 ± 1.2**	64.8 ± 1.2	1	**28.8 ± 0.7**	23.5 ± 0.5	1
	PONIX	69.8 ± 1.2	1.8 ± 0.7	10	**82.5 ± 1.0**	**-0.6 ± 1.1**	10	75.2 ± 1.6	**0.0 ± 0.0**	10	25.5 ± 1.0	**-0.4 ± 0.6**	10
π_0	ER†	56.3 ± 2.8	**-6.8 ± 1.8**	1	**87.9 ± 0.8**	1.9 ± 1.9	1	89.7 ± 1.1	**-1.1 ± 3.4**	1	73.2 ± 1.5	**-0.5 ± 1.0**	1
	TAIL	63.9 ± 2.1	-0.2 ± 0.2	10	81.6 ± 0.0	**-0.1 ± 0.1**	10	**93.0 ± 0.5**	0.0 ± 0.0	10	**92.3 ± 0.6**	0.1 ± 0.1	10
	Seq	64.9 ± 1.0	57.3 ± 0.1	1	81.3 ± 0.9	73.6 ± 0.4	1	88.7 ± 0.8	73.3 ± 1.1	1	89.2 ± 1.9	82.1 ± 2.2	1
	PONIX	**65.8 ± 2.5**	0.2 ± 0.5	10	73.9 ± 1.7	6.6 ± 1.4	10	90.7 ± 0.8	-0.3 ± 0.6	10	89.1 ± 1.8	0.0 ± 0.2	10
SmolVLA	TAIL	**56.7 ± 1.3**	**-0.3 ± 0.2**	10	**76.9 ± 3.0**	**0.0 ± 0.0**	10	64.8 ± 2.4	**0.0 ± 0.0**	10	80.5 ± 1.0	**0.0 ± 0.2**	10
	Seq	42.8 ± 2.8	40.7 ± 2.5	1	69.6 ± 0.9	66.4 ± 1.2	1	73.1 ± 1.4	59.2 ± 1.5	1	79.9 ± 1.1	76.8 ± 0.4	1
	PONIX	45.9 ± 0.8	0.3 ± 0.8	12	73.4 ± 1.9	0.1 ± 0.3	10	**77.2 ± 1.6**	**0.0 ± 0.0**	10	**83.8 ± 1.4**	**0.0 ± 0.1**	10

		Meta-World											
		Easy			Medium			Hard			Very Hard		
		SR ↑	NBT ↓	$\lvert\mathcal{E}\rvert$ ↓	SR ↑	NBT ↓	$\lvert\mathcal{E}\rvert$ ↓	SR ↑	NBT ↓	$\lvert\mathcal{E}\rvert$ ↓	SR ↑	NBT ↓	$\lvert\mathcal{E}\rvert$ ↓
π_0	TAIL	**83.7 ± 0.6**	**0.0 ± 0.1**	28	**63.0 ± 1.8**	**-0.8 ± 1.1**	11	59.8 ± 2.0	**0.6 ± 1.1**	6	56.5 ± 4.6	0.0 ± 3.1	5
	Seq	77.9 ± 0.5	67.9 ± 0.7	1	46.2 ± 2.1	40.4 ± 1.5	1	**69.9 ± 2.2**	59.1 ± 1.9	1	54.8 ± 1.8	42.4 ± 1.7	1
	PONIX	79.6 ± 0.5	**0.0 ± 0.4**	28	60.2 ± 1.4	0.4 ± 1.3	11	60.1 ± 2.9	12.6 ± 1.2	5	**61.7 ± 3.4**	**-0.3 ± 0.7**	5
SmolVLA	TAIL	**84.9 ± 0.5**	**0.0 ± 0.3**	28	**65.8 ± 3.9**	0.5 ± 0.6	11	68.3 ± 2.6	**0.7 ± 0.9**	6	**59.3 ± 2.6**	1.0 ± 1.3	5
	Seq	83.5 ± 0.3	73.1 ± 0.7	1	54.5 ± 2.4	48.5 ± 1.5	1	**73.3 ± 3.8**	60.8 ± 3.2	1	56.4 ± 1.4	48.0 ± 2.2	1
	PONIX	82.7 ± 0.5	**0.0 ± 0.6**	28	55.5 ± 4.5	**-0.4 ± 2.5**	11	73.1 ± 3.6	15.6 ± 1.2	5	54.1 ± 0.2	**-1.7 ± 1.1**	5

Table 1: **Continual-learning performance on LIBERO and Meta-World.** Gray rows denote baselines using additional supervision or memory: TAIL uses oracle task identities; ER uses task boundary signals and exemplars. †Published ER results (Liu et al., 2026a) use more trainable parameters than our LoRA configuration.

most settings. PONIX achieves NBT within two percentage points of TAIL in 16 of 20 settings and lower NBT than Seq in every setting. The success-rate trajectories illustrate sustained retention; for example, π_0 preserves earlier task performance on LIBERO-Object while Seq repeatedly loses it (Figures 50 and 48). **(3) Expert allocation usually tracks the task count.** PONIX's final expert count equals the task count in 17 of 20 settings. On Meta-World, the counts match in every suite except Hard for both models. Training and evaluation activation plots show task-specific selection on LIBERO-Object with π_0 (Figures 51 and 52).

These results support continual VLA adaptation with acquisition and retention approaching oracle-task-ID adaptation in many settings, without task identities, boundary signals, or stored exemplars.

5.3 Task Revisitation

Can PONIX reuse experts when tasks recur while maintaining acquisition and retention without explicit recurrence labels? We use SmolVLA on LIBERO, splitting each task's $\mathcal{D}_\mathcal{T}$ into two disjoint halves of whole trajectories. We present the first halves for all tasks, then the second halves in the same task order, giving $M = 2K = 20$ collections. Each visit receives half the original per-task training budget. Algorithm 2 specifies the visit protocol, and Table 16 lists the trajectory counts per split.

To specify revisit behavior absent from TAIL's original protocol, we introduce two variants. *Instruction TAIL* uses instructions as task IDs. *Boundary TAIL* initializes an adapter at every visit boundary, retains all adapters, and evaluates with the latest adapter for each instruction. LIBERO's design makes instruction lookup equivalent to task-ID lookup in these suites (allocation rules in Table 17). Table 2 yields three findings: **(1) Acquisition remains comparable to the TAIL variants.** PONIX's FWT is within five percentage points of both variants. **(2) Forgetting remains close to the TAIL variants.** PONIX's NBT is within two percentage points of both variants on every suite, ranging from -1.3 to 0.4 points. **(3) Expert reuse limits growth under revisitation.** PONIX retains 10–11

Method	LIBERO															
	10				Spatial				Object				Goal			
	FWT ↑	SR ↑	NBT ↓	$\lvert\mathcal{E}\rvert$ ↓	FWT ↑	SR ↑	NBT ↓	$\lvert\mathcal{E}\rvert$ ↓	FWT ↑	SR ↑	NBT ↓	$\lvert\mathcal{E}\rvert$ ↓	FWT ↑	SR ↑	NBT ↓	$\lvert\mathcal{E}\rvert$ ↓
Boundary TAIL	**44.1 ± 1.2**	**46.4 ± 1.6**	-0.9 ± 1.4	20	74.4 ± 1.0	**75.6 ± 2.1**	-0.3 ± 0.6	20	74.6 ± 1.5	75.7 ± 1.5	-0.5 ± 0.2	20	82.8 ± 1.3	83.6 ± 2.1	0.2 ± 0.9	20
Instruction TAIL	39.5 ± 0.3	39.2 ± 1.1	-0.1 ± 0.5	10	**75.4 ± 0.6**	73.0 ± 1.8	0.4 ± 0.4	10	**76.4 ± 0.4**	**80.4 ± 1.8**	**-1.1 ± 0.4**	10	82.2 ± 0.6	82.2 ± 0.3	0.2 ± 0.4	10
Seq	40.6 ± 2.2	34.7 ± 3.1	37.4 ± 2.0	1	61.9 ± 1.4	60.5 ± 2.6	57.8 ± 1.5	1	61.9 ± 0.2	49.9 ± 0.6	52.7 ± 0.3	1	75.9 ± 0.1	76.2 ± 1.6	70.5 ± 0.1	1
PONIX	39.4 ± 2.3	42.9 ± 2.0	**-1.1 ± 0.5**	11	72.0 ± 0.9	75.3 ± 0.9	**-1.3 ± 0.9**	10	74.4 ± 0.6	75.3 ± 0.6	0.4 ± 0.5	10	**83.6 ± 1.5**	**88.8 ± 0.5**	**-1.3 ± 0.2**	10

Table 2: **Task revisitation with SmolVLA on LIBERO.** Gray rows denote TAIL variants using boundary signals or instruction-based task identities.

experts; the training activations and expert-count curves in Appendix C show experts being selected again on revisits.

These results show that PONIX reuses experts without explicit recurrence labels, maintaining acquisition and retention comparable to the TAIL variants.

5.4 Real-Robot Experiments

Can PONIX on a real robot, distinguish different tasks sharing an instruction, and group the same task across different instructions? We evaluate SmolVLA with PONIX on an SO-101 arm with two RGB cameras (execution settings in Table 18). The stream contains $K = 3$ tasks across $M = 4$ collections: "Put the pen into the cup," "Put the tape in the bowl" with an uncovered bowl, "Put the pen in the cup," and "Put the tape in the bowl" with a covered bowl. The two pen collections share a task, $\mathcal{T}_3 = \mathcal{T}_1$. Table 19 gives the episode assignments, and Figure 131 shows each collection's initial scene. Success requires releasing the object inside the container, except for the covered-bowl condition, where we score lid opening because of the longer task horizon; tape placement is not required. The success-rate and activation plots in Figures 132–135 support two findings: **(1) Identical instructions can select different experts.** PONIX allocates separate experts to the uncovered and covered bowl conditions despite their identical instruction. At their respective acquisition checkpoints, evaluation selects expert E_1 for the uncovered bowl and E_2 for the covered bowl. **(2) Different instructions for the same task share an expert.** Both pen collections train expert E_0, which is also selected for both wordings at the final checkpoint. After reusing this expert for the paraphrased instruction, success on the original instruction increases from 50% to 80%, reaching 90% at the final checkpoint (Table 20). Relative to its initial acquisition success of 60%, the original collection has a first-visit NBT of -13.3 percentage points, indicating improvement in the previously learned task. Thus, four collections require only three experts.

These results demonstrate real-robot adaptation with expert separation and reuse that reflect the task beyond instruction wording.

6 Conclusion

We formulated pseudo-task-free continual learning for VLAs, exploiting task-homogeneous mini-batches without task identities or boundary signals. Our exemplar-free method, PONIX, combines frozen multimodal representations with online statistics to allocate and reuse action-side LoRA experts. Experiments on LIBERO and Meta-World show acquisition and retention approaching oracle-task-ID adaptation in many settings. Under task revisitation, PONIX maintains performance comparable to TAIL variants while using approximately half as many experts as Boundary TAIL. Real-robot experiments further demonstrate that expert allocation can distinguish tasks sharing an instruction and group the same task across different wordings. These results support continual VLA adaptation through inferred expert allocation and reuse, without globally annotated task identities or stored demonstrations.

AI USE STATEMENT

We used OpenAI Codex to assist with manuscript writing, create figures from author-provided hand-drawn sketches, and check the consistency of the manuscript's descriptions with the implementation code. The authors take responsibility for the final manuscript and figures, including all AI-assisted content.

ETHICS STATEMENT

This work studies continual adaptation of vision-language-action models for robotic manipulation in simulation and on an SO-101 robot arm. Incorrect expert selection or policy errors can produce unintended physical actions, and performance on the evaluated tasks does not establish safe operation in other environments. Deployment therefore requires application-specific safety evaluation, human oversight, and appropriate safeguards. The absence of stored exemplars in PONIX is not a formal privacy guarantee for the learned parameters or routing statistics.

REPRODUCIBILITY STATEMENT

The source code is included in the Supplementary Material. Section 3 defines the learning setting and available supervision, while Section 4 and Algorithm 1 describe expert routing, expansion, and adaptation. Section 5 describes the benchmarks, baseline protocols, evaluation metrics, task-revisitation construction, and real-robot evaluation setup. Appendices A–D provide the detailed implementation and experimental settings for the simulation, task-revisitation, and real-robot experiments. We will publicly release the trained model weights, real-robot dataset, and evaluation videos upon acceptance.

REFERENCES

Rahaf Aljundi, Lucas Caccia, Eugene Belilovsky, Massimo Caccia, Min Lin, Laurent Charlin, and Tinne Tuytelaars. Online continual learning with maximally interfered retrieval. In *Advances in Neural Information Processing Systems*, volume 32, 2019a. URL https://papers.neurips.cc/paper/9357-online-continual-learning-with-maximal-interfered-retrieval.pdf.

Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019b.

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 *9th Annual Conference on Robot Learning*, 2025a. URL https://openreview.net/forum?id=vlhoswksBO.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025b. doi: 10.15607/RSS.2025.XXI.010.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and SIMONE CALDERARA. Dark experience for general continual learning: a strong, simple baseline. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 15920–15930. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/b704ea2c39778f07c617f6b7ce480e9e-Paper.pdf.

T. F. Chan, G. H. Golub, and R. J. LeVeque. Updating formulae and a pairwise algorithm for computing sample variances. In H. Caussinus, P. Ettinger, and R. Tomassone (eds.), *COMPSTAT 1982 5th Symposium held at Toulouse 1982*, pp. 30–41, Heidelberg, 1982. Physica-Verlag HD. ISBN 978-3-642-51461-6.

Mohamed Elsayed and A. Rupam Mahmood. Addressing loss of plasticity and catastrophic forgetting in continual learning. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=sKPzAXoylB.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022. URL http://jmlr.org/papers/v23/21-0998.html.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1126–1135. PMLR, 2017. URL https://proceedings.mlr.press/v70/finn17a.html.

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.

Jiaheng Hu, Jay Shim, Chen Tang, Yoonchang Sung, Bo Liu, Peter Stone, and Roberto Martín-Martín. Simple recipe works: Vision-language-action models are natural continual learners with reinforcement learning. *Reinforcement Learning Journal*, 7, 2026.

Qiongfeng Shi Jiaqi Wang, Zhou Fang and Yi Zhou. Continual skill learning via gradient-informed skill subspace adaptation. In *Proceedings of the Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*, 2026.

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 *8th Annual Conference on Robot Learning*, 2024. URL https://openreview.net/forum?id=ZMnD6QZAE6.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. doi: 10.1073/pnas.1611835114. URL https://www.pnas.org/doi/abs/10.1073/pnas.1611835114.

Daehee Lee, Minjong Yoo, Woo Kyung Kim, Wonje Choi, and Honguk Woo. Incremental learning of retrievable skills for efficient continual task adaptation. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 17286–17312. Curran Associates, Inc., 2024. doi: 10.52202/079017-0549. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/1f0832859514e53a0e4f229fc9b3a4a2-Paper-Conference.pdf.

Soochan Lee, Junsoo Ha, Dongsu Zhang, and Gunhee Kim. A neural dirichlet process mixture model for task-free continual learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SJxSOJStPr.

Yuheng Lei, Sitong Mao, Shunbo Zhou, Hongyuan Zhang, Xuelong Li, and Ping Luo. Dynamic mixture of progressive parameter-efficient expert library for lifelong robot learning. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=MHVBrjS8cG.

Hongbo Li, Sen Lin, Lingjie Duan, Yingbin Liang, and Ness Shroff. Theory on mixture-of-experts in continual learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=7XgKAabsPp.

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 (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 44776–44791. Curran Associates, Inc., 2023. doi: 10.52202/075280-1939. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/8c3c666820ea055a77726d66fc7d447f-Paper-Datasets_and_Benchmarks.pdf.

Huihan Liu, Changyeon Kim, Bo Liu, Minghuan Liu, and Yuke Zhu. Pretrained vision-language-action models are surprisingly resistant to forgetting in continual learning. In *Forty-third International Conference on Machine Learning*, 2026a. URL https://openreview.net/forum?id=VzdSHEab4G.

Yuan Liu, Haoran Li, Shuai Tian, Yuxing Qin, Yuhui Chen, Yupeng Zheng, Yongzhen Huang, and Dongbin Zhao. Towards long-lived robots: Continual learning vla models via reinforcement fine-tuning, 2026b. URL https://arxiv.org/abs/2602.10503.

Zuxin Liu, Jesse Zhang, Kavosh Asadi, Yao Liu, Ding Zhao, Shoham Sabach, and Rasool Fakoor. TAIL: Task-specific adapters for imitation learning with large pretrained models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=RRayv1ZPN3.

NVIDIA, :, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. Gr00t n1: An open foundation model for generalist humanoid robots, 2025. URL https://arxiv.org/abs/2503.14734.

Fuli Qiao and Mehrdad Mahdavi. Merge before forget: A single loRA continual learning via continual merging. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=i1Rj7yU6eF.

Ralf Römer, Yi Zhang, Yuming Li, and Angela P. Schoellig. Clare: Continual learning for vision-language-action models via autonomous adapter routing and expansion. *IEEE Robotics and Automation Letters*, 11(7):8331–8338, 2026. doi: 10.1109/LRA.2026.3693992.

Noam Shazeer, *Azalia Mirhoseini, *Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=B1ckMDqlg.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. Smolvla: A vision-language-action model for affordable and efficient robotics, 2025. URL https://arxiv.org/abs/2506.01844.

Weikang Wan, Yifeng Zhu, Rutav Shah, and Yuke Zhu. Lotus: Continual imitation learning for robot manipulation through unsupervised skill discovery. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 537–544, 2024. doi: 10.1109/ICRA57147.2024.10611129.

Huiyi Wang, Haodong Lu, Lina Yao, and Dong Gong. Self-expansion of pre-trained models with mixture of adapters for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10087–10098, June 2025.

Xudong Wang, Zebin Han, Zhiyu Liu, Gan Li, Jiahua Dong, Baichen Liu, Lianqing Liu, and Zhi Han. Lifelong language-conditioned robotic manipulation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18629–18637, 2026. doi: 10.1609/aaai.v40i22.38930. URL https://ojs.aaai.org/index.php/AAAI/article/view/38930.

Yixiao Wang, Yifei Zhang, Mingxiao Huo, Thomas Tian, Xiang Zhang, Yichen Xie, Chenfeng Xu, Pengliang Ji, Wei Zhan, Mingyu Ding, and Masayoshi Tomizuka. Sparse diffusion policy: A sparse, reusable, and flexible policy for robot learning. In *8th Annual Conference on Robot Learning*, 2024. URL https://openreview.net/forum?id=zeYaLS2tw5.

B. P. Welford. Note on a method for calculating corrected sums of squares and products. *Technometrics*, 4(3):419–420, 1962. doi: 10.1080/00401706.1962.10490022. URL https://www.tandfonline.com/doi/abs/10.1080/00401706.1962.10490022.

Fei Ye and Adrian G. Bors. Task-free continual learning via online discrepancy distance learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 23675–23688. Curran Associates, Inc., 2022. doi: 10.52202/068431-1720. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/95c6ae3f3393786203a4b6dcb9df1036-Paper-Conference.pdf.

Fanqi Yu, Matteo Tiezzi, Tommaso Apicella, Cigdem Beyan, and Vittorio Murino. Lifelong imitation learning with multimodal latent replay and incremental adjustment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6740–6749, June 2026.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura (eds.), *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 1094–1100. PMLR, 30 Oct–01 Nov 2020. URL https://proceedings.mlr.press/v100/yu20a.html.

Qixin Zeng, Shuo Zhang, Hongyin Zhang, Renjie Wang, Han Zhao, Libang Zhao, Runze Li, Donglin Wang, and Chao Huang. Crl-vla: Continual vision-language-action learning, 2026. URL https://arxiv.org/abs/2602.03445.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3987–3995. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/zenke17a.html.

Chen Zeno, Itay Golan, Elad Hoffer, and Daniel Soudry. Task-agnostic continual learning using online variational bayes with fixed-point updates. *Neural Computation*, 33(11):3139–3177, 2021. doi: 10.1162/neco_a_01430. URL https://doi.org/10.1162/neco_a_01430.

A Implementation and Reporting Details

A.1 Evaluation Metrics

We report acquisition success (SR), forward transfer (FWT), and negative backward transfer (NBT). Our SR and NBT definitions follow Liu et al. (2026a), extended to repeated task visits. We consider a stream of M collections covering K distinct tasks. Collection $\mathcal{C}_i$ contains demonstrations of task $\mathcal{T}_i$. The success rate on $\mathcal{T}_i$ after training through $\mathcal{C}_j$ is denoted by $c_{i,j} \in [0, 1]$. The set $\mathcal{L} = \{i : \mathcal{T}_i \neq \mathcal{T}_j \text{ for all } j > i\}$ contains the last-visit index for each distinct task, so $|\mathcal{L}| = K$. The metrics are computed as follows:

$$\mathrm{SR} = \frac{1}{K} \sum_{i \in \mathcal{L}} c_{i,i}, \quad \mathrm{FWT} = \frac{1}{M} \sum_{i=1}^{M} c_{i,i}, \quad \mathrm{NBT} = \frac{1}{M} \sum_{i=1}^{M-1} \frac{1}{M-i} \sum_{j=i+1}^{M} (c_{i,i} - c_{i,j}). \tag{6}$$

SR averages success after each distinct task's last visit, whereas FWT averages success after every collection. When no task is revisited, $M = K$ and $\mathrm{FWT} = \mathrm{SR}$. Our FWT does not measure zero-shot performance or the area under a within-task learning curve. NBT measures the average drop in acquisition success at later checkpoints, including revisits. Negative values indicate improvement. We also report the final number of experts, $|\mathcal{E}|$.

A.2 Training and Routing Configuration

Tables 3–6 summarize the saved training configurations. Each run uses training seed 1000. Mini-batches sample shuffled frames from the current collection; normalization statistics remain fixed across the stream. Image augmentation and EMA are disabled. The pretrained backbone is frozen, and LoRA has rank 16, scale 16, no bias adaptation, and no additional fully trained modules.

Table 3: Optimization and action-generation settings. Real uses SmolVLA; Split inherits the simulation SmolVLA settings.

Setting	SmolVLA	π_0	GR00T	Real
Learning rate	2×10^{-3}	2×10^{-3}	2×10^{-4}	3×10^{-4}
AdamW β_1	0.9	0.9	0.9	0.9
AdamW β_2	0.95	0.95	0.999	0.95
AdamW ε	10^{-8}	10^{-8}	10^{-8}	10^{-8}
Weight decay	10^{-10}	10^{-2}	10^{-5}	10^{-10}
Gradient clipping norm	10	1	1	10
Batch size	128	128	128	128
Gradient accumulation	1	1	1	1
Learning-rate schedule	Constant	Constant	Constant	Constant
Warm-up updates	0	0	0	0
Observation frames per input	1	1	1	1
Policy image size	512^2	224^2	256^2	512^2
Generated action chunk	50	50	40	50
Simulation actions per replan	10	10	10	—
Drop partial training batches	No	No	No	Yes
Checkpoint interval (updates)	1,000	1,000	1,000	1,000

SmolVLA and π_0 use mean/standard-deviation state and action normalization; GR00T uses its embodiment-specific preprocessing. The trainer uses eight data workers and logs every 200 updates. SmolVLA and π_0 use ten denoising steps; GR00T retains its base checkpoint's inference-step default.

Table 4: Pretrained policies and adapter placement. The action-side attention query/value projections and listed projections receive LoRA.

Policy	Base checkpoint	Additional LoRA projections
SmolVLA	SmolVLA base	Action input/output; action-time MLP
π_0	π_0 base	Action input/output; action-time MLP
GR00T	GR00T N1.7 (3B)	Action-head timestep/output; embodiment-specific action encoder/decoder

Table 5: PONIX routing settings. The saved key `sigma_init` is the additive scatter prior λ, not a fixed variance floor.

Setting	SmolVLA	π_0	GR00T	Real
Scatter prior λ	1	0.5	0.5	1
Numerical ε	10^{-6}	10^{-6}	10^{-6}	10^{-6}
VLM layer index	0	0	0	-1 (last)
Modalities	Vision+text	Vision+text	Vision+text	Vision+text
Visual tokens pooled	All views	All views	All views	Front only
Inference selection	Top-1	Top-1	Top-1	Top-1
Inference expansion/statistic updates	Off	Off	Off	Off

Table 6: Saved optimizer and scheduler reset flags at collection boundaries.

Exp.	Model	Method	Benchmark	Optimizer	Schedule
Main	GR00T	PONIX	LIBERO	No	No
Main	GR00T	Seq	LIBERO	Yes	Yes
Main	GR00T	TAIL	LIBERO	Yes	Yes
Main	π_0	PONIX	Both	No	No
Main	π_0	Seq	Both	No	No
Main	π_0	TAIL	Both	Yes	Yes
Main	SmolVLA	PONIX	Both	No	No
Main	SmolVLA	Seq	Meta-World	No	No
Main	SmolVLA	Seq	LIBERO	Yes	Yes
Main	SmolVLA	TAIL	Both	Yes	Yes
Split	SmolVLA	Boundary TAIL	LIBERO	Yes	Yes
Split	SmolVLA	PONIX	LIBERO	No	No
Split	SmolVLA	Instruction TAIL	LIBERO	Yes	Yes
Split	SmolVLA	Seq	LIBERO	No	No

PONIX separately resets its scheduler on expert expansion; the configured learning rate is constant.

Reading the plots. T0, T1, ... index stream positions; E0, E1, ... index expert creation. Grey vertical lines mark collection transitions. Simulation SR curves show the mean and sample standard deviation across evaluation seeds 1000, 2000, and 3000, with 50 episodes per seed and task/checkpoint. These bands describe evaluation variability, not independently trained models. Training heatmaps show fractions of routed mini-batches obtained by differencing cumulative selection counts; captions give the available logging resolution. Simulation evaluation heatmaps use logged activation fractions and separate seed panels. White–blue represents zero–one activation; grey denotes unavailable entries. Expert indices are not pooled across runs. No SR values are imputed.

B Simulation Experiments

We use the LIBERO and Meta-World demonstration datasets. Table 7 reports the actual eligible demonstrations, rather than assuming 50 per task. All previously encountered tasks are evaluated after every 10,000-update collection. The following figures cover every model–suite–method combination; PONIX additionally includes training and final evaluation activation.

Table 7: Simulation stream sizes and evaluation budgets per method and model.

Suite	Tasks	Demos	Updates	Eval. episodes
LIBERO-Spatial	10	432	100k	8,250
LIBERO-Object	10	454	100k	8,250
LIBERO-Goal	10	428	100k	8,250
LIBERO-10	10	379	100k	8,250
Meta-World Easy	28	1396	280k	60,900
Meta-World Medium	11	525	110k	9,900
Meta-World Hard	6	296	60k	3,150
Meta-World Very Hard	5	248	50k	2,250

Each task receives 10,000 updates. Evaluation totals are $3 imes 50 imes K(K+1)/2$: 50 episodes for each seen task at each boundary, under three evaluation seeds.

B.1 TASK ORDERS

Table 8: LIBERO-Spatial task order. T0 is the first collection.

Index	Instruction
T0	pick up the black bowl between the plate and the ramekin and place it on the plate
T1	pick up the black bowl next to the ramekin and place it on the plate
T2	pick up the black bowl from table center and place it on the plate
T3	pick up the black bowl on the cookie box and place it on the plate
T4	pick up the black bowl in the top drawer of the wooden cabinet and place it on the plate
T5	pick up the black bowl on the ramekin and place it on the plate
T6	pick up the black bowl next to the cookie box and place it on the plate
T7	pick up the black bowl on the stove and place it on the plate
T8	pick up the black bowl next to the plate and place it on the plate
T9	pick up the black bowl on the wooden cabinet and place it on the plate

Table 9: LIBERO-Object task order. T0 is the first collection.

Index	Instruction
T0	pick up the alphabet soup and place it in the basket
T1	pick up the cream cheese and place it in the basket
T2	pick up the salad dressing and place it in the basket
T3	pick up the bbq sauce and place it in the basket
T4	pick up the ketchup and place it in the basket
T5	pick up the tomato sauce and place it in the basket
T6	pick up the butter and place it in the basket
T7	pick up the milk and place it in the basket
T8	pick up the chocolate pudding and place it in the basket
T9	pick up the orange juice and place it in the basket

Table 10: LIBERO-Goal task order. T0 is the first collection.

Index	Instruction
T0	open the middle drawer of the cabinet
T1	put the bowl on the stove
T2	put the wine bottle on top of the cabinet
T3	open the top drawer and put the bowl inside
T4	put the bowl on top of the cabinet
T5	push the plate to the front of the stove
T6	put the cream cheese in the bowl
T7	turn on the stove
T8	put the bowl on the plate
T9	put the wine bottle on the rack

Table 11: LIBERO-10 task order. T0 is the first collection.

Index	Instruction
T0	put both the alphabet soup and the tomato sauce in the basket
T1	put both the cream cheese box and the butter in the basket
T2	turn on the stove and put the moka pot on it

Index	Instruction
T3	put the black bowl in the bottom drawer of the cabinet and close it
T4	put the white mug on the left plate and put the yellow and white mug on the right plate
T5	pick up the book and place it in the back compartment of the caddy
T6	put the white mug on the plate and put the chocolate pudding to the right of the plate
T7	put both the alphabet soup and the cream cheese box in the basket
T8	put both moka pots on the stove
T9	put the yellow and white mug in the microwave and close it

Table 12: Meta-World Easy task order. T0 is the first collection.

Index	Instruction
T0	Press a button
T1	Press a button from the top
T2	Bypass a wall and press a button from the top
T3	Bypass a wall and press a button
T4	Push a button on the coffee machine
T5	Rotate a dial 180 degrees
T6	Close a door with a revolving joint
T7	Lock the door by rotating the lock clockwise
T8	Open a door with a revolving joint
T9	Unlock the door by rotating the lock counter-clockwise
T10	Push and close a drawer
T11	Open a drawer
T12	Rotate the faucet clockwise
T13	Rotate the faucet counter-clockwise
T14	Press a handle down
T15	Press a handle down sideways
T16	Pull a handle up
T17	Pull a handle up sideways
T18	Pull a lever down 90 degrees
T19	Slide a plate into a cabinet
T20	Get a plate from the cabinet
T21	Get a plate from the cabinet sideways
T22	Slide a plate into a cabinet sideways
T23	Reach a goal position
T24	Bypass a wall and reach a goal
T25	Push and close a window
T26	Push and open a window
T27	Unplug a peg sideways

Table 13: Meta-World Medium task order. T0 is the first collection.

Index	Instruction
T0	Dunk the basketball into the basket
T1	Grasp the puck from one bin and place it into another bin
T2	Grasp the cover and close the box with it
T3	Pull a mug from a coffee machine
T4	Push a mug under a coffee machine
T5	Hammer a screw on the wall
T6	Insert a peg sideways
T7	Bypass a wall and push a puck to a goal
T8	Kick a soccer into the goal

Index	Instruction
T9	Sweep a puck off the table
T10	Sweep a puck into a hole

Table 14: Meta-World Hard task order. T0 is the first collection.

Index	Instruction
T0	Pick up a nut and place it onto a peg
T1	Insert the gripper into a hole
T2	Pick up a puck from a hole
T3	Pick and place a puck to a goal
T4	Push the puck to a goal
T5	Push the puck to a goal

Table 15: Meta-World Very Hard task order. T0 is the first collection.

Index	Instruction
T0	Pick and place a puck onto a shelf
T1	Pick a nut out of a peg
T2	Grasp a stick and pull a box with the stick
T3	Grasp a stick and push a box using the stick
T4	Pick a puck, bypass a wall and place the puck

B.2 SmolVLA on LIBERO-Spatial

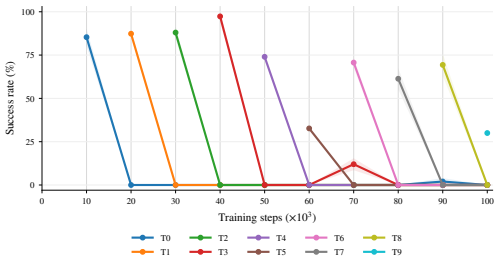


Figure 3: Seq: task-wise success rates, SmolVLA on LIBERO-Spatial. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

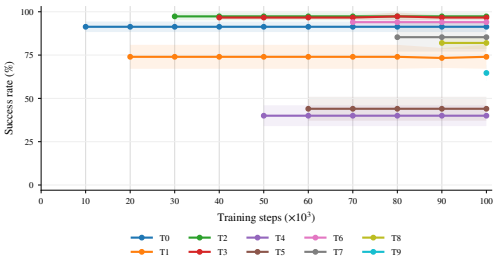


Figure 4: TAIL: task-wise success rates, SmolVLA on LIBERO-Spatial. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

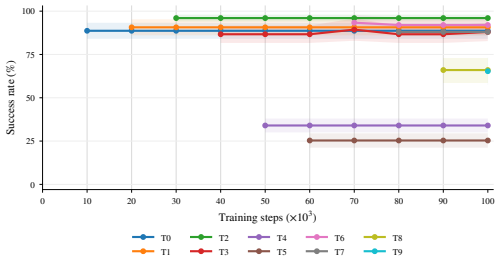


Figure 5: PONIX: task-wise success rates, SmolVLA on LIBERO-Spatial. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

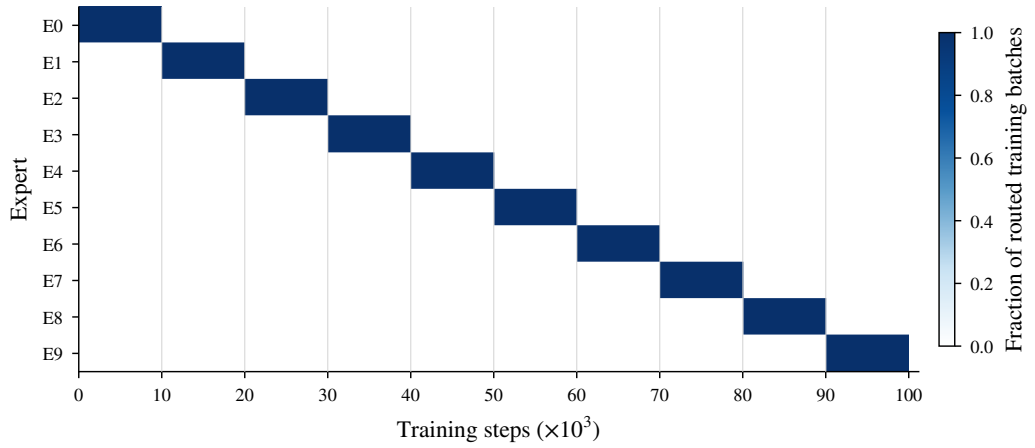


Figure 6: PONIX training activation, SmolVLA on LIBERO-Spatial. Fractions of routed batches in 10,000-update intervals; expert IDs follow creation order.

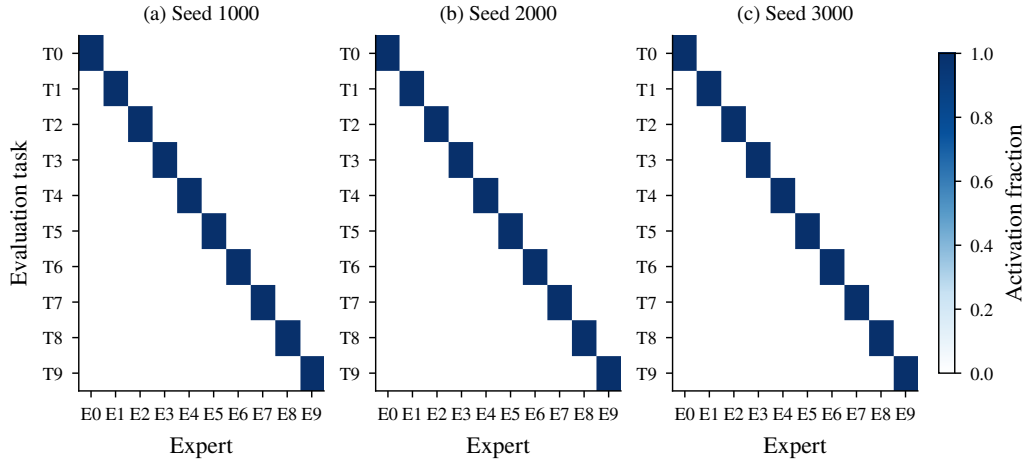


Figure 7: PONIX final evaluation activation, SmolVLA on LIBERO-Spatial. Panels correspond to separate evaluation seeds.

B.3 SmolVLA on LIBERO-Object

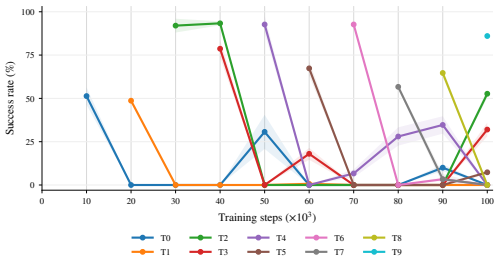


Figure 8: Seq: task-wise success rates, SmolVLA on LIBERO-Object. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

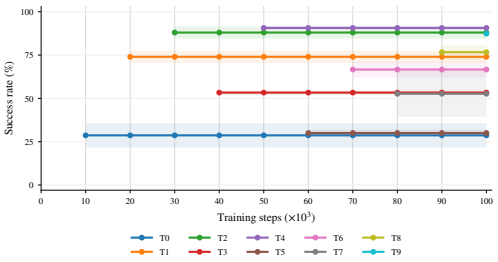


Figure 9: TAIL: task-wise success rates, SmolVLA on LIBERO-Object. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

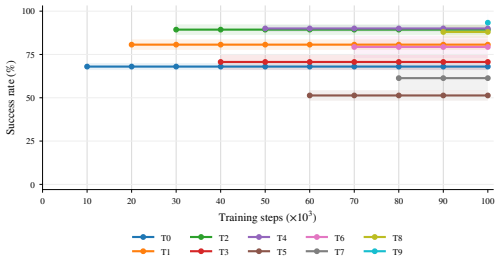


Figure 10: PONIX: task-wise success rates, SmolVLA on LIBERO-Object. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

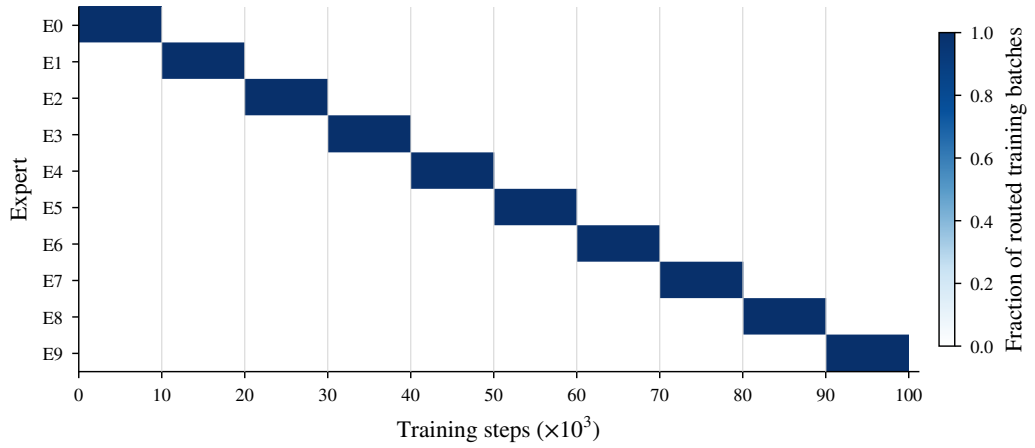


Figure 11: PONIX training activation, SmolVLA on LIBERO-Object. Fractions of routed batches in 10,000-update intervals; expert IDs follow creation order.

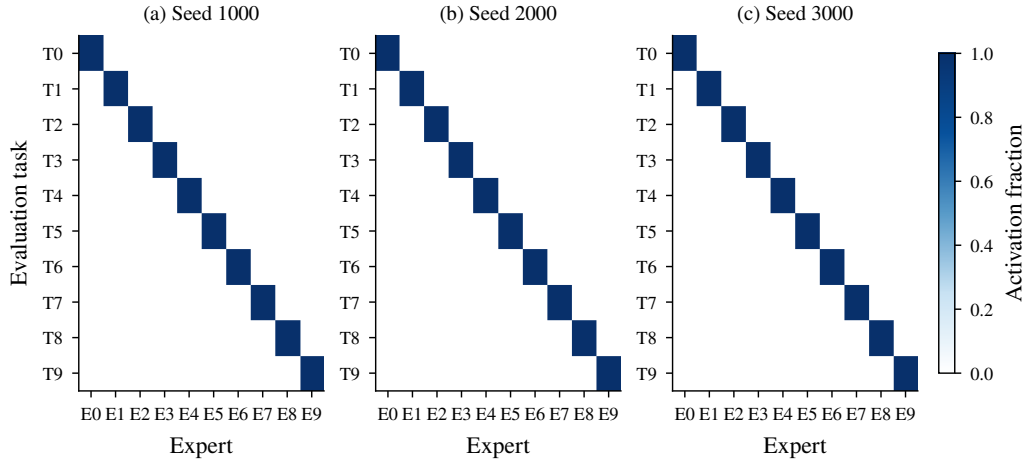


Figure 12: PONIX final evaluation activation, SmolVLA on LIBERO-Object. Panels correspond to separate evaluation seeds.

B.4 SmolVLA on LIBERO-Goal

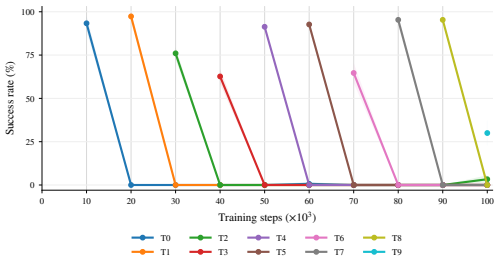


Figure 13: Seq: task-wise success rates, SmolVLA on LIBERO-Goal. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

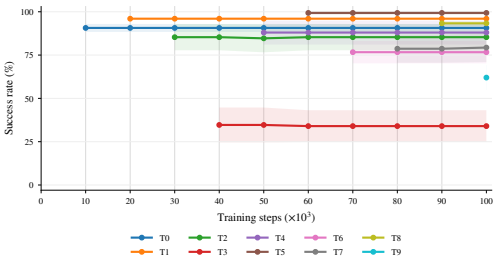


Figure 14: TAIL: task-wise success rates, SmolVLA on LIBERO-Goal. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

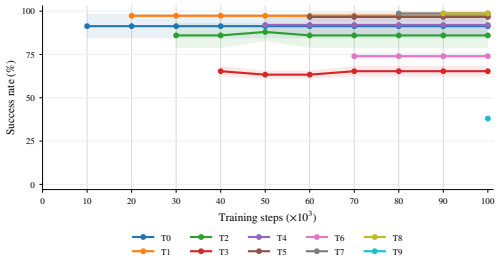


Figure 15: PONIX: task-wise success rates, SmolVLA on LIBERO-Goal. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

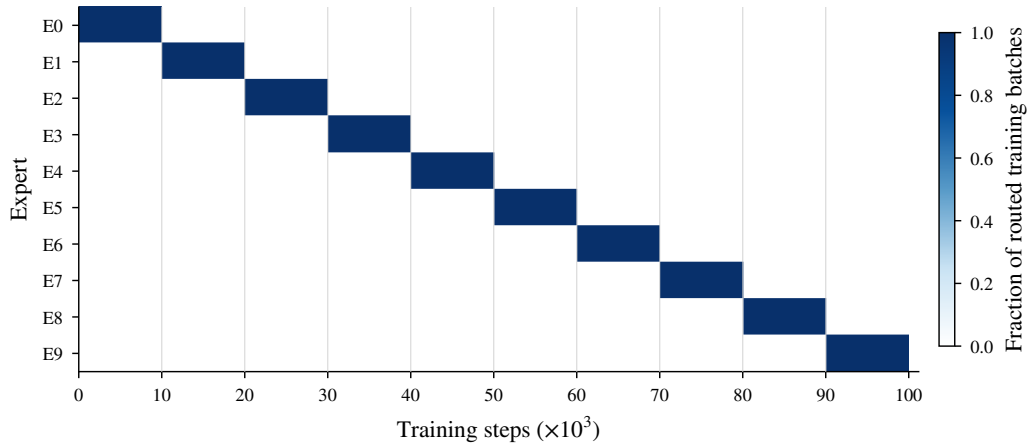


Figure 16: PONIX training activation, SmolVLA on LIBERO-Goal. Fractions of routed batches in 10,000-update intervals; expert IDs follow creation order.

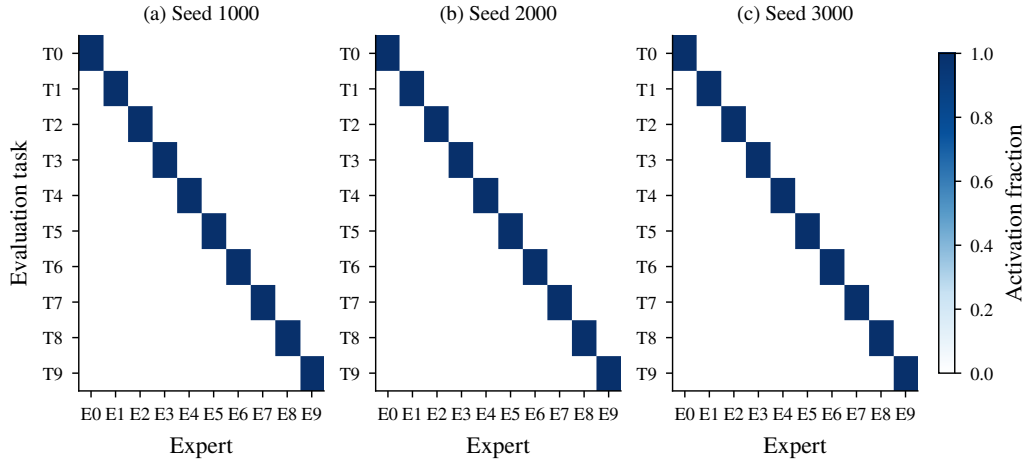


Figure 17: PONIX final evaluation activation, SmolVLA on LIBERO-Goal. Panels correspond to separate evaluation seeds.

B.5 SmolVLA on LIBERO-10

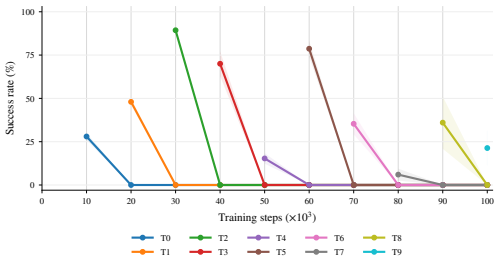


Figure 18: Seq: task-wise success rates, SmolVLA on LIBERO-10. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

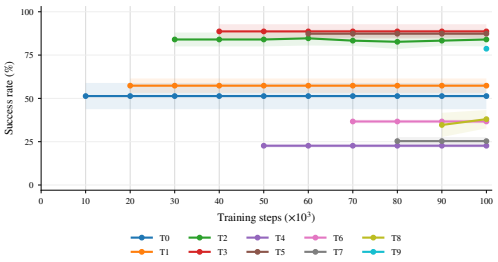


Figure 19: TAIL: task-wise success rates, SmolVLA on LIBERO-10. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

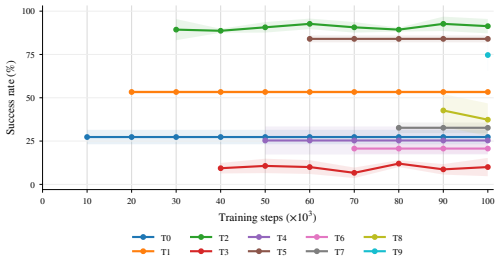


Figure 20: PONIX: task-wise success rates, SmolVLA on LIBERO-10. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

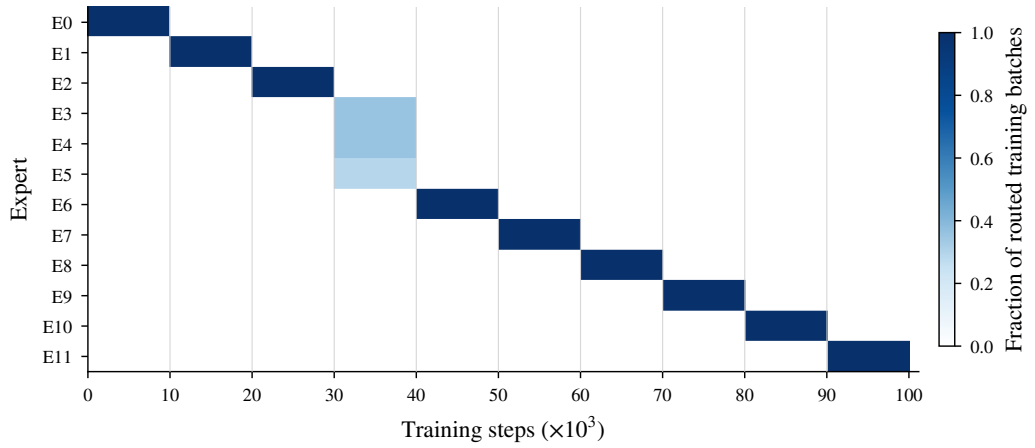


Figure 21: PONIX training activation, SmolVLA on LIBERO-10. Fractions of routed batches in 10,000-update intervals; expert IDs follow creation order.

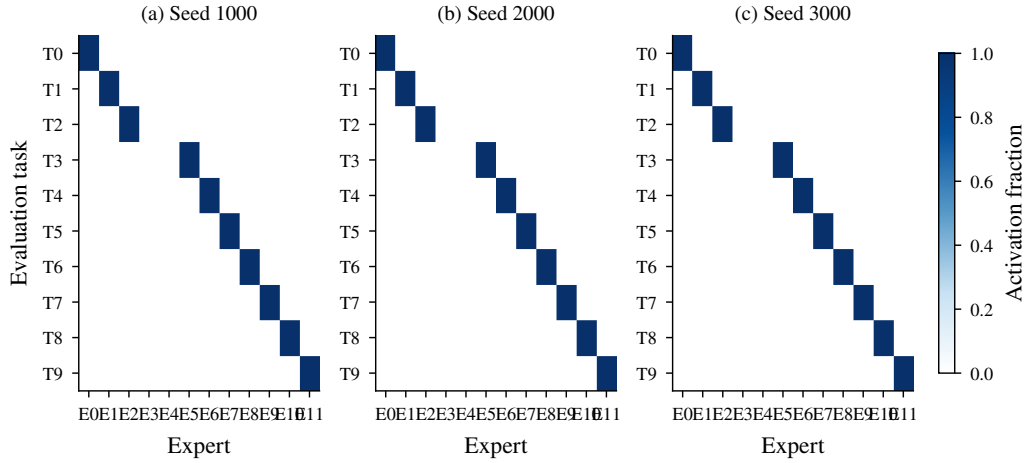


Figure 22: PONIX final evaluation activation, SmolVLA on LIBERO-10. Panels correspond to separate evaluation seeds.

B.6 SmolVLA on Meta-World Easy

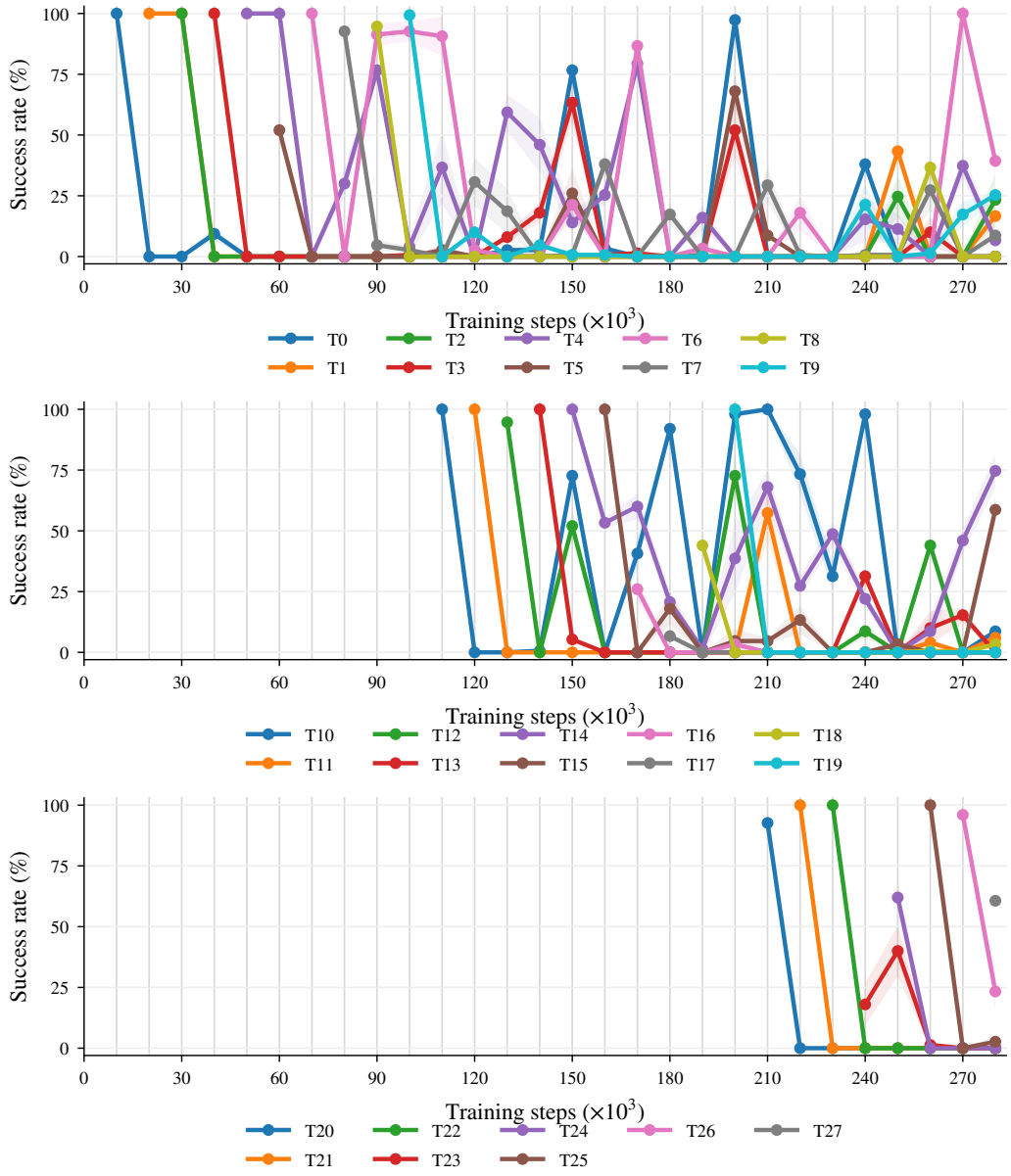


Figure 23: Seq: task-wise success rates, SmolVLA on Meta-World Easy. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

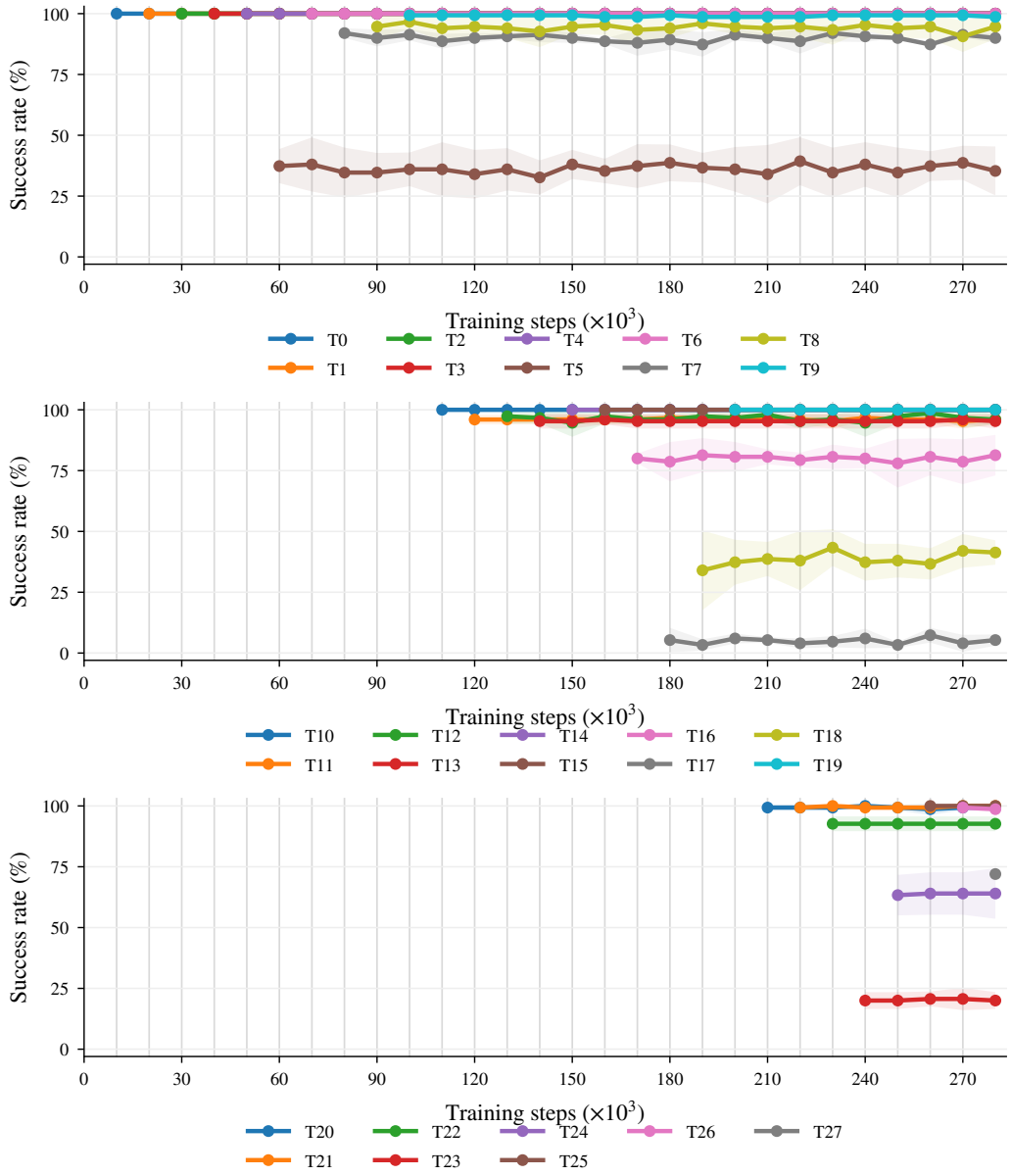


Figure 24: TAIL: task-wise success rates, SmolVLA on Meta-World Easy. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

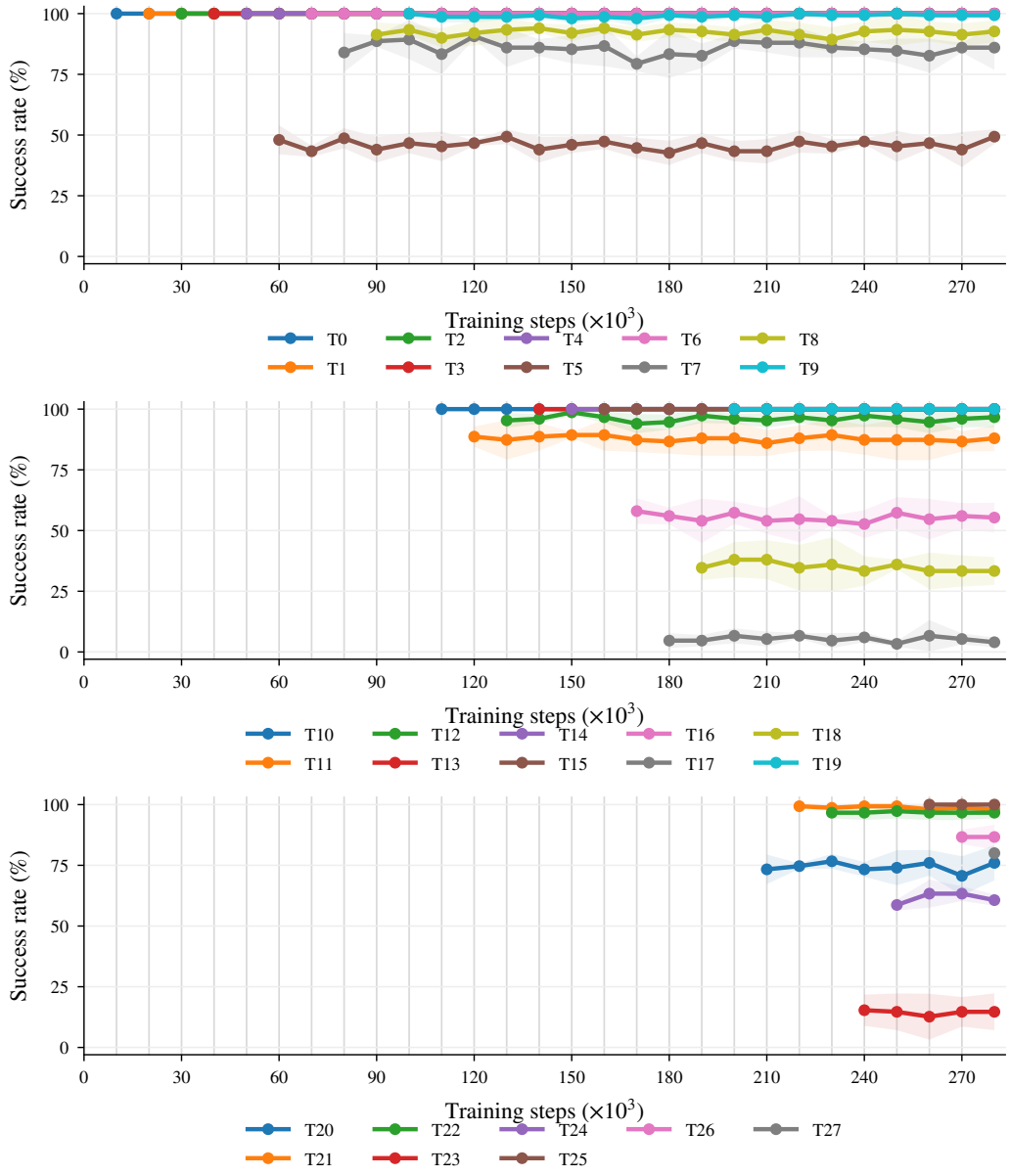


Figure 25: PONIX: task-wise success rates, SmolVLA on Meta-World Easy. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

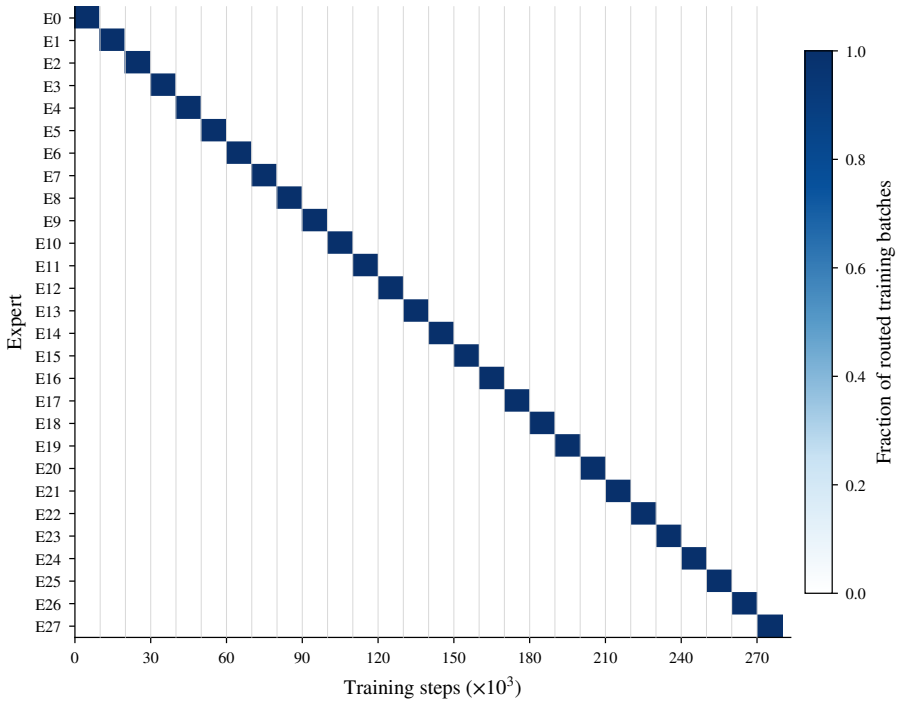


Figure 26: PONIX training activation, SmolVLA on Meta-World Easy. Fractions of routed batches in 10,000-update intervals; expert IDs follow creation order.

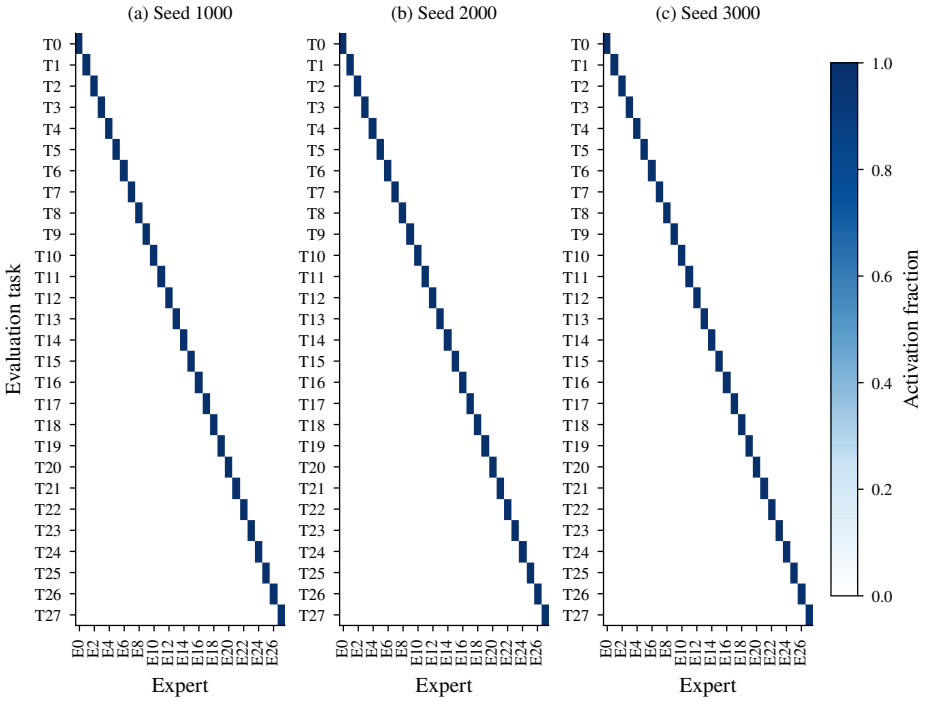


Figure 27: PONIX final evaluation activation, SmolVLA on Meta-World Easy. Panels correspond to separate evaluation seeds.

B.7 SmolVLA on Meta-World Medium

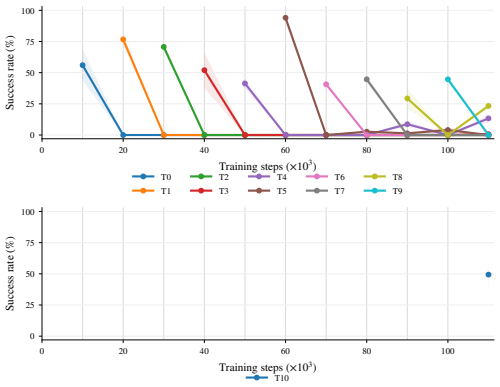


Figure 28: Seq: task-wise success rates, SmolVLA on Meta-World Medium. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

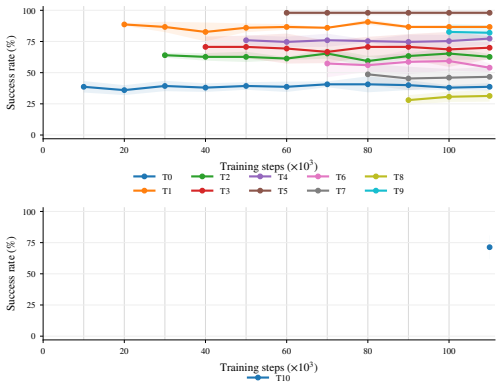


Figure 29: TAIL: task-wise success rates, SmolVLA on Meta-World Medium. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

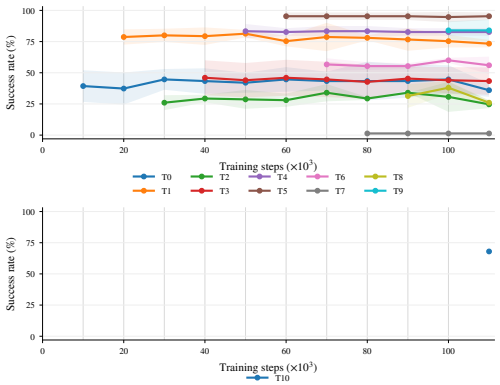


Figure 30: PONIX: task-wise success rates, SmolVLA on Meta-World Medium. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

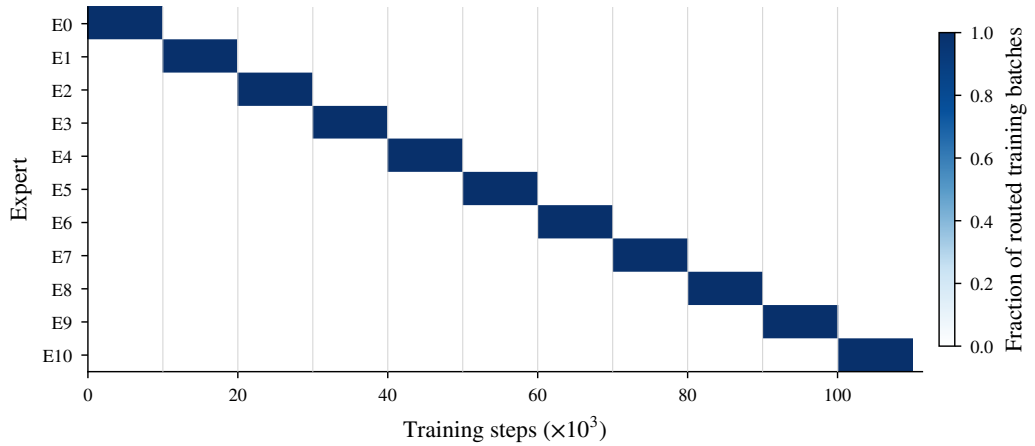


Figure 31: PONIX training activation, SmolVLA on Meta-World Medium. Fractions of routed batches in 10,000-update intervals; expert IDs follow creation order.

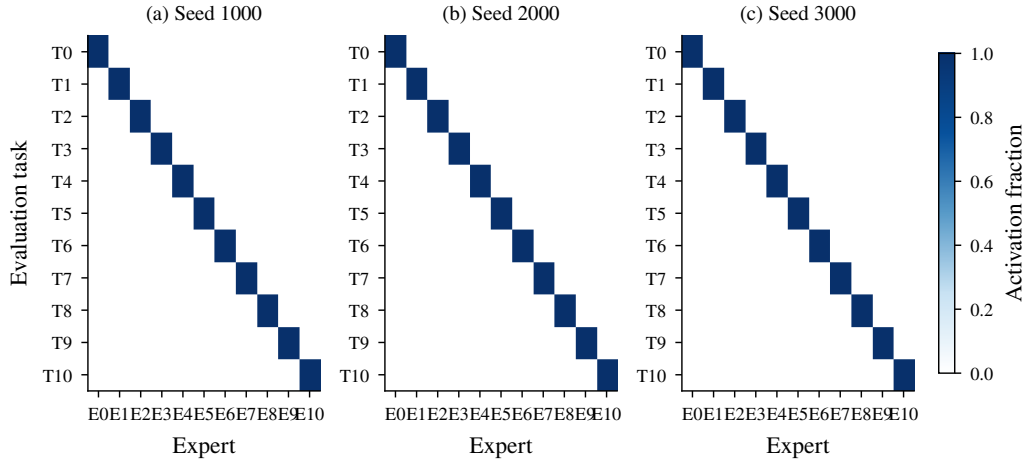


Figure 32: PONIX final evaluation activation, SmolVLA on Meta-World Medium. Panels correspond to separate evaluation seeds.

B.8 SmolVLA on Meta-World Hard

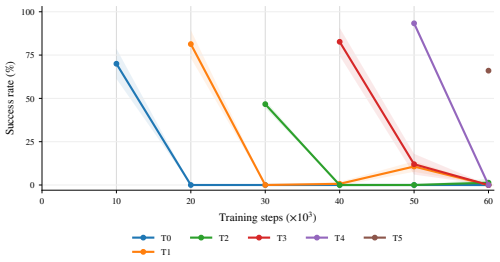


Figure 33: Seq: task-wise success rates, SmolVLA on Meta-World Hard. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

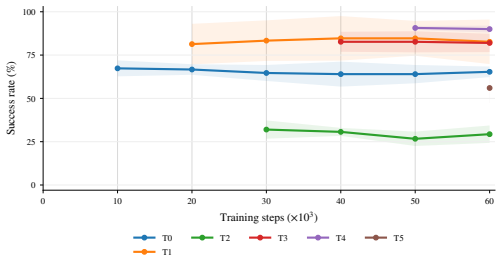


Figure 34: TAIL: task-wise success rates, SmolVLA on Meta-World Hard. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

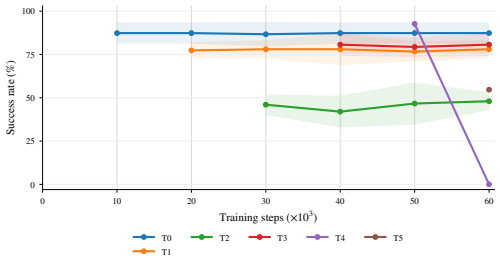


Figure 35: PONIX: task-wise success rates, SmolVLA on Meta-World Hard. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

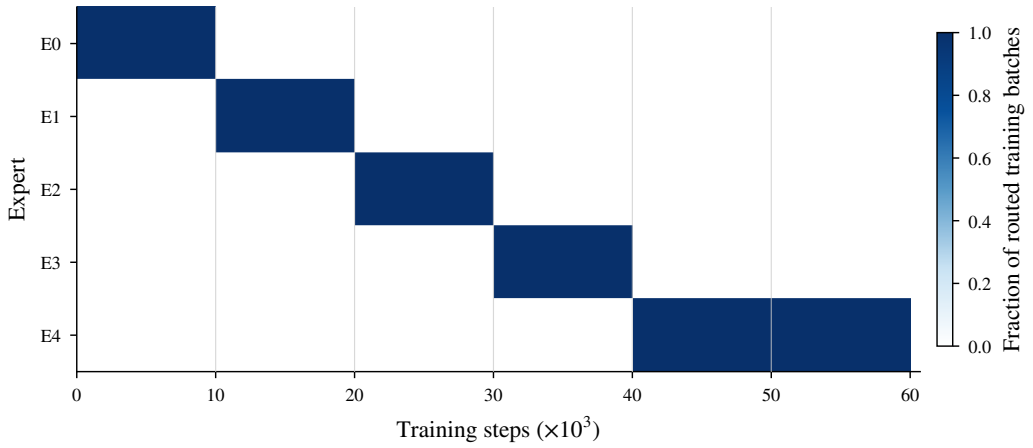


Figure 36: PONIX training activation, SmolVLA on Meta-World Hard. Fractions of routed batches in 10,000-update intervals; expert IDs follow creation order.

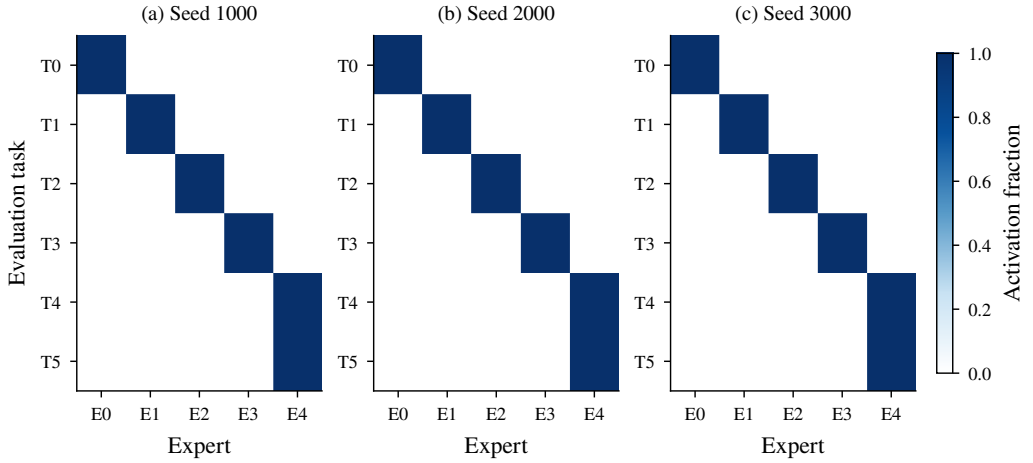


Figure 37: PONIX final evaluation activation, SmolVLA on Meta-World Hard. Panels correspond to separate evaluation seeds.

B.9 SmolVLA on Meta-World Very Hard

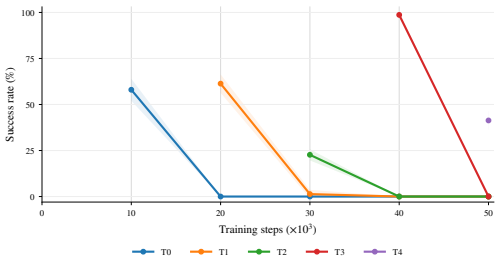


Figure 38: Seq: task-wise success rates, SmolVLA on Meta-World Very Hard. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

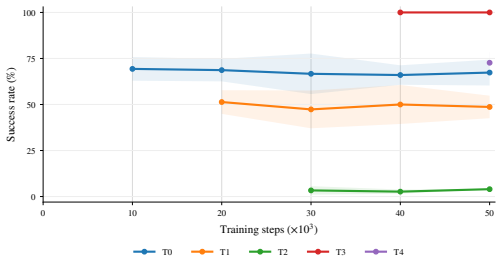


Figure 39: TAIL: task-wise success rates, SmolVLA on Meta-World Very Hard. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

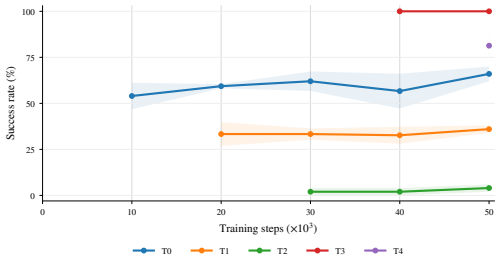


Figure 40: PONIX: task-wise success rates, SmolVLA on Meta-World Very Hard. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

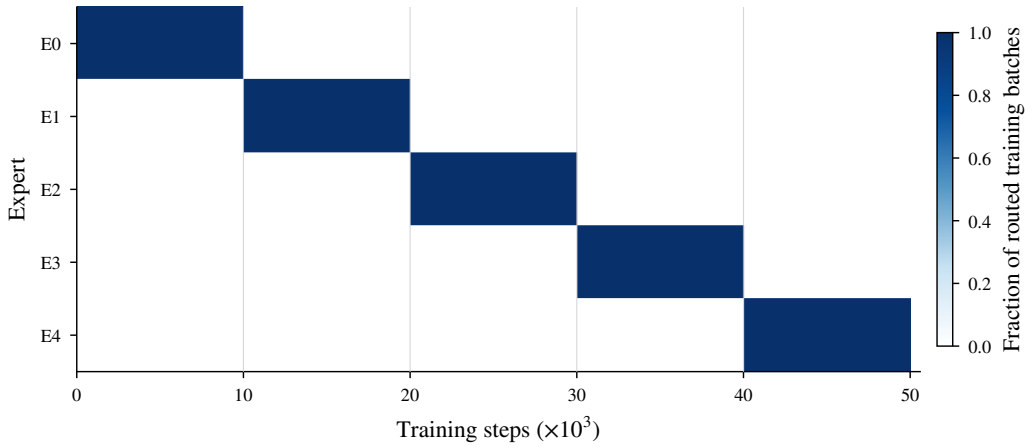


Figure 41: PONIX training activation, SmolVLA on Meta-World Very Hard. Fractions of routed batches in 10,000-update intervals; expert IDs follow creation order.

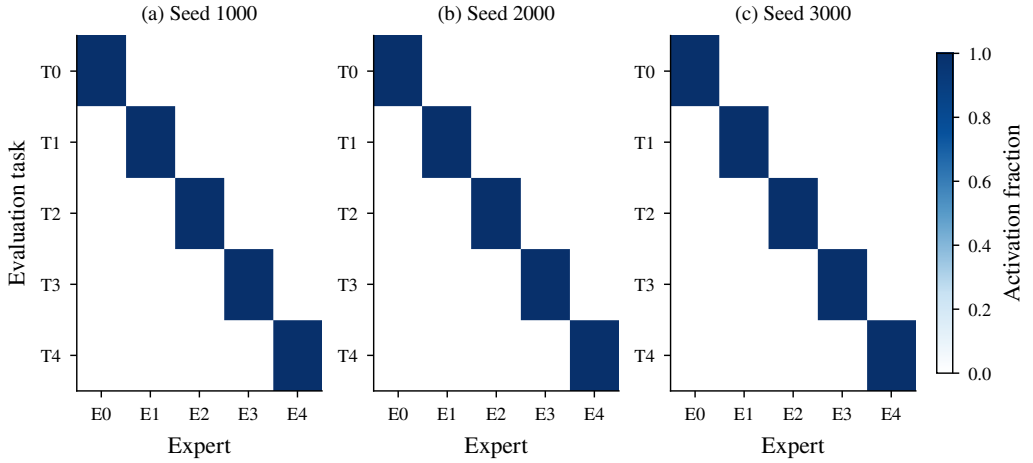


Figure 42: PONIX final evaluation activation, SmolVLA on Meta-World Very Hard. Panels correspond to separate evaluation seeds.

B.10 π_0 ON LIBERO-SPATIAL

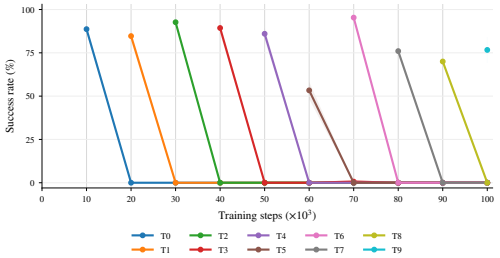


Figure 43: Seq: task-wise success rates, π_0 on LIBERO-Spatial. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

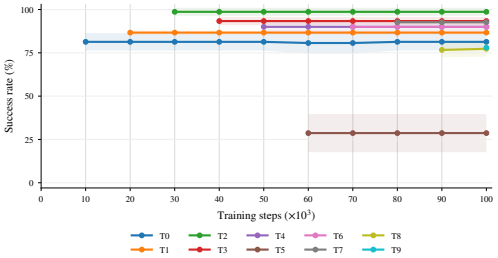


Figure 44: TAIL: task-wise success rates, π_0 on LIBERO-Spatial. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

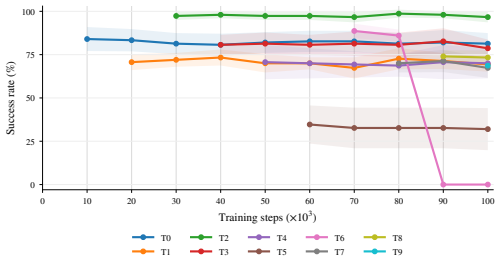


Figure 45: PONIX: task-wise success rates, π_0 on LIBERO-Spatial. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

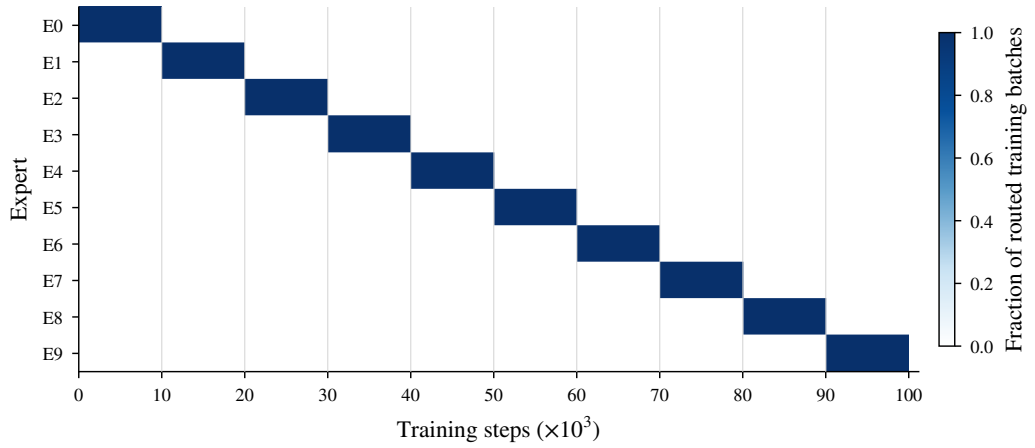


Figure 46: PONIX training activation, π_0 on LIBERO-Spatial. Fractions of routed batches in 200–10,000-update intervals; expert IDs follow creation order.

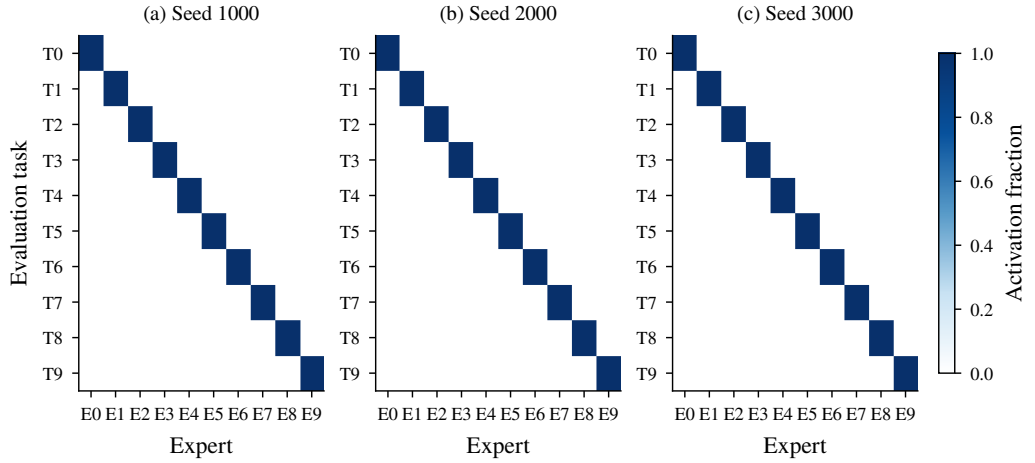


Figure 47: PONIX final evaluation activation, π_0 on LIBERO-Spatial. Panels correspond to separate evaluation seeds.

B.11 π_0 ON LIBERO-OBJECT

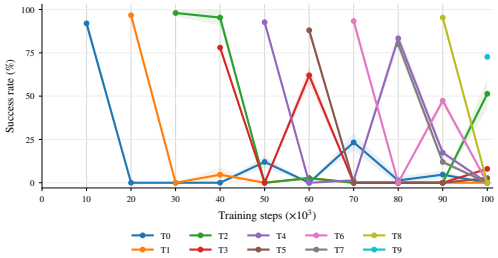


Figure 48: Seq: task-wise success rates, π_0 on LIBERO-Object. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

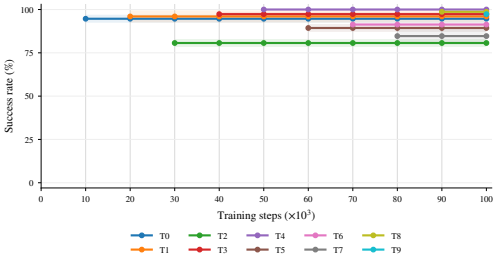


Figure 49: TAIL: task-wise success rates, π_0 on LIBERO-Object. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

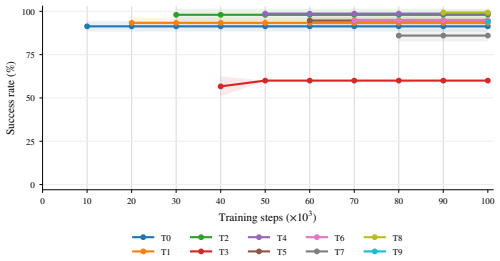


Figure 50: PONIX: task-wise success rates, π_0 on LIBERO-Object. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

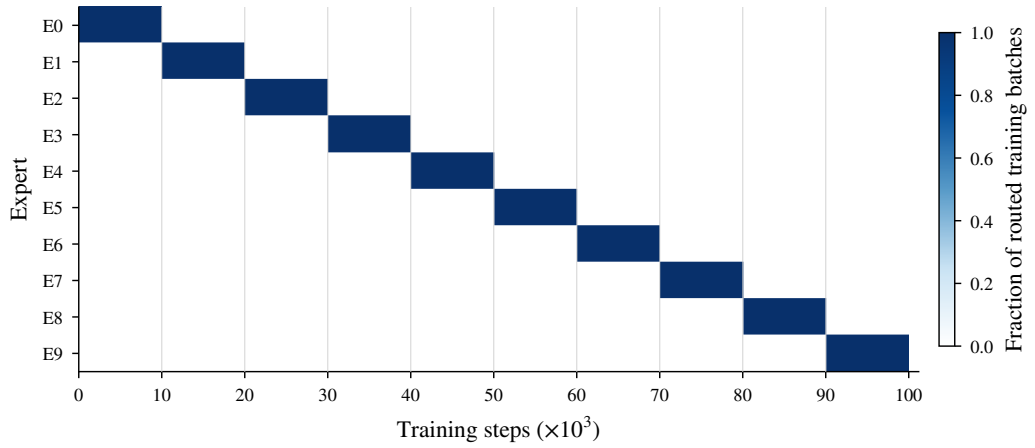


Figure 51: PONIX training activation, π_0 on LIBERO-Object. Fractions of routed batches in 200–10,000-update intervals; expert IDs follow creation order.

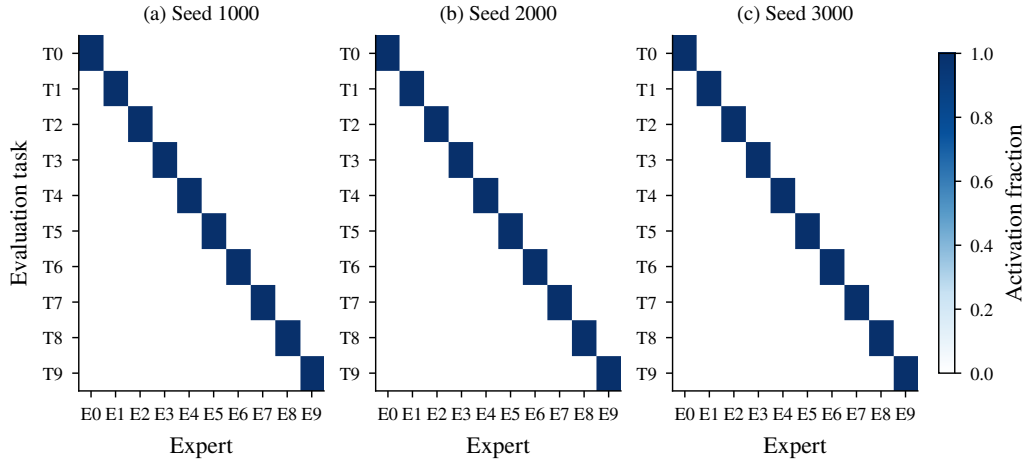


Figure 52: PONIX final evaluation activation, π_0 on LIBERO-Object. Panels correspond to separate evaluation seeds.

B.12 π_0 ON LIBERO-GOAL

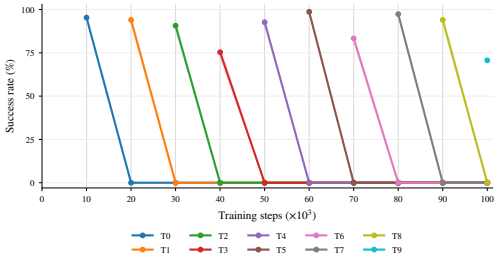


Figure 53: Seq: task-wise success rates, π_0 on LIBERO-Goal. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

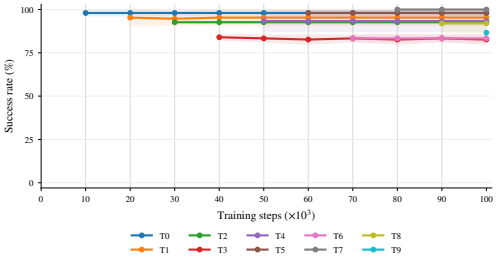


Figure 54: TAIL: task-wise success rates, π_0 on LIBERO-Goal. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

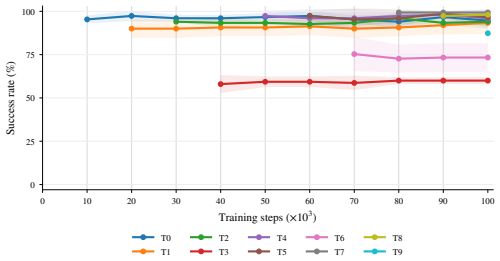


Figure 55: PONIX: task-wise success rates, π_0 on LIBERO-Goal. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

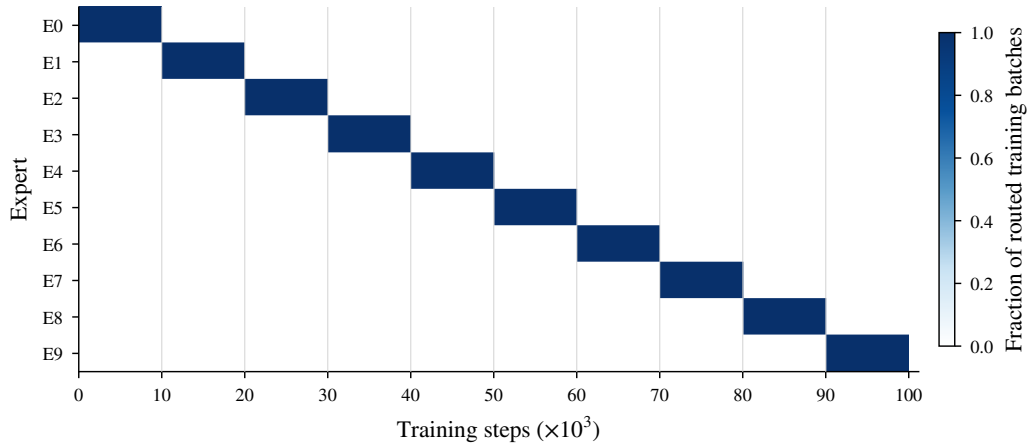


Figure 56: PONIX training activation, π_0 on LIBERO-Goal. Fractions of routed batches in 200–10,000-update intervals; expert IDs follow creation order.

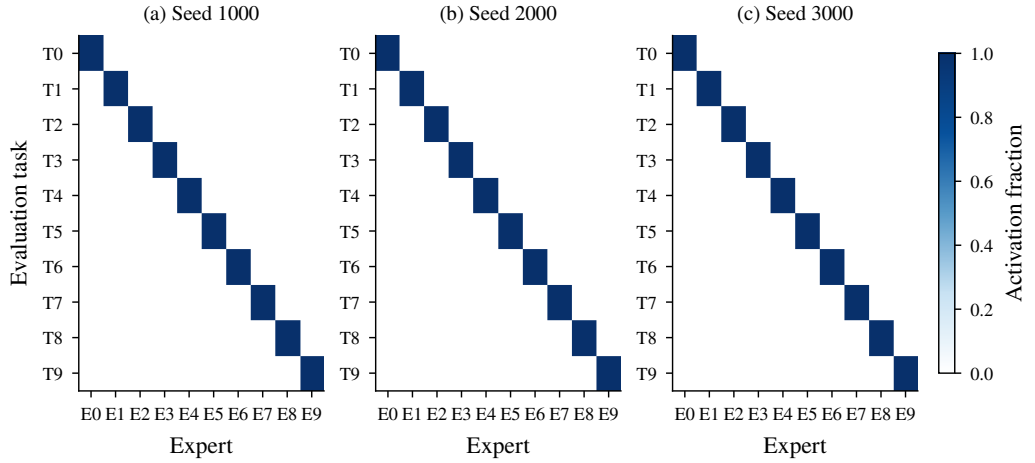


Figure 57: PONIX final evaluation activation, π_0 on LIBERO-Goal. Panels correspond to separate evaluation seeds.

B.13 π_0 ON LIBERO-10

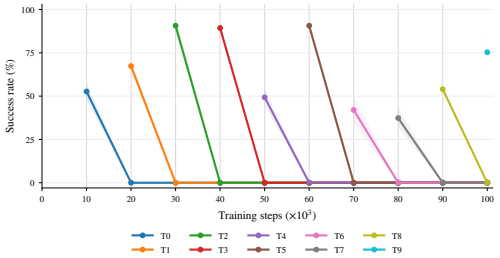


Figure 58: Seq: task-wise success rates, π_0 on LIBERO-10. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

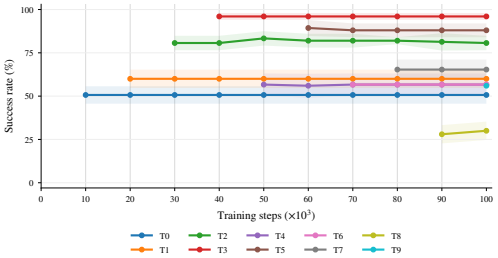


Figure 59: TAIL: task-wise success rates, π_0 on LIBERO-10. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

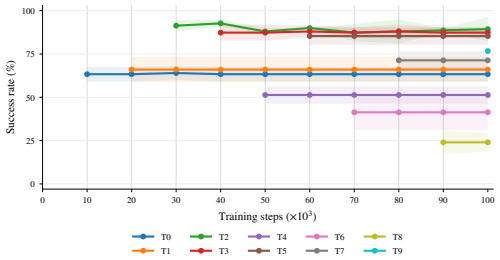


Figure 60: PONIX: task-wise success rates, π_0 on LIBERO-10. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

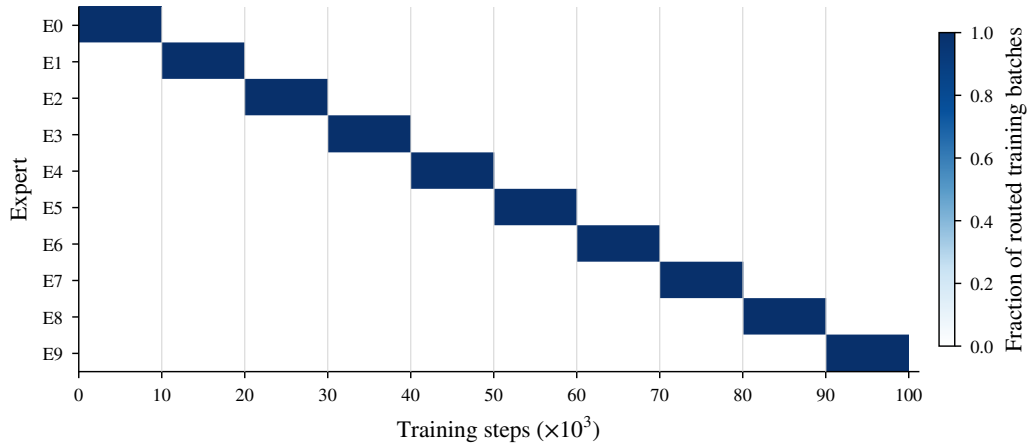


Figure 61: PONIX training activation, π_0 on LIBERO-10. Fractions of routed batches in 10,000-update intervals; expert IDs follow creation order.

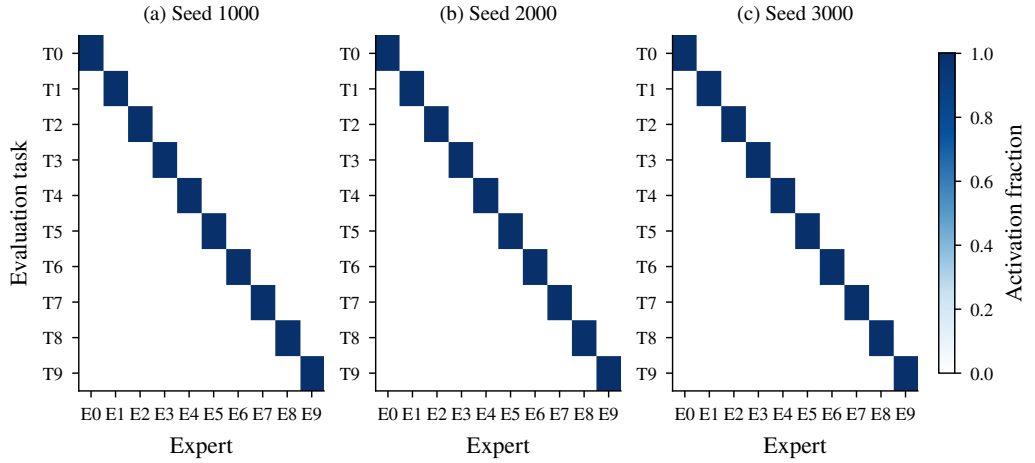


Figure 62: PONIX final evaluation activation, π_0 on LIBERO-10. Panels correspond to separate evaluation seeds.

B.14 π_0 ON META-WORLD EASY

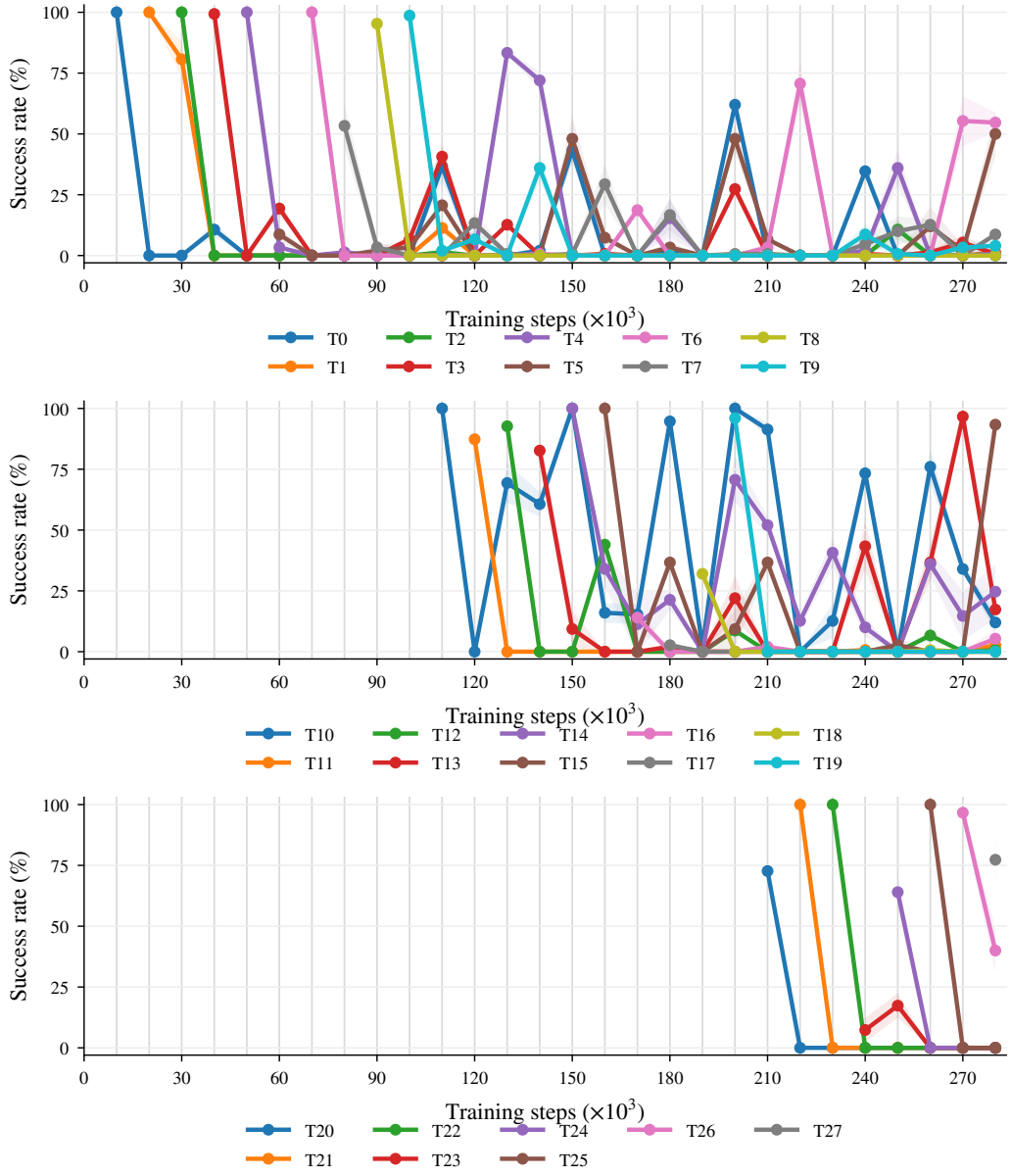


Figure 63: Seq: task-wise success rates, π_0 on Meta-World Easy. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

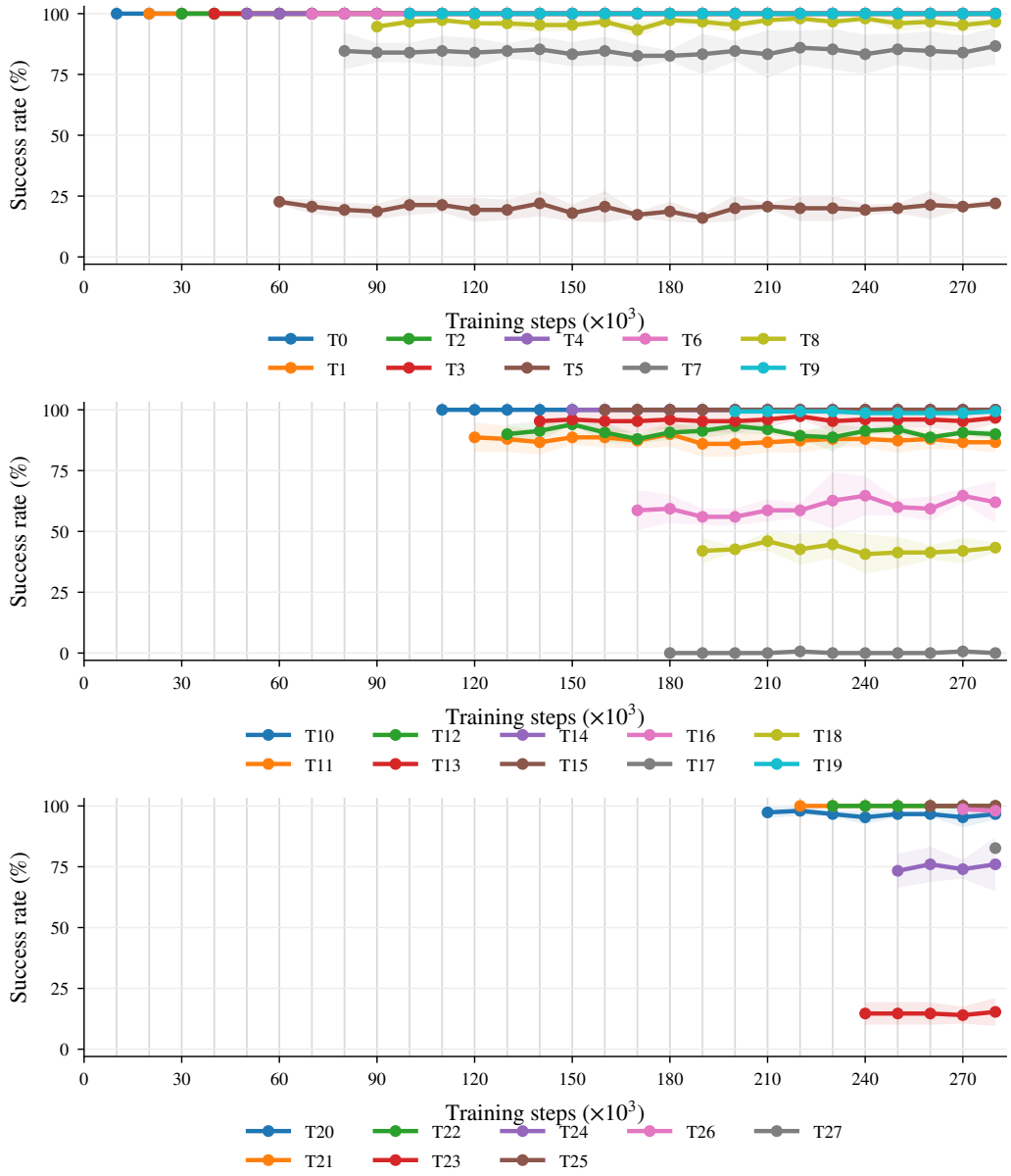


Figure 64: TAIL: task-wise success rates, π_0 on Meta-World Easy. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

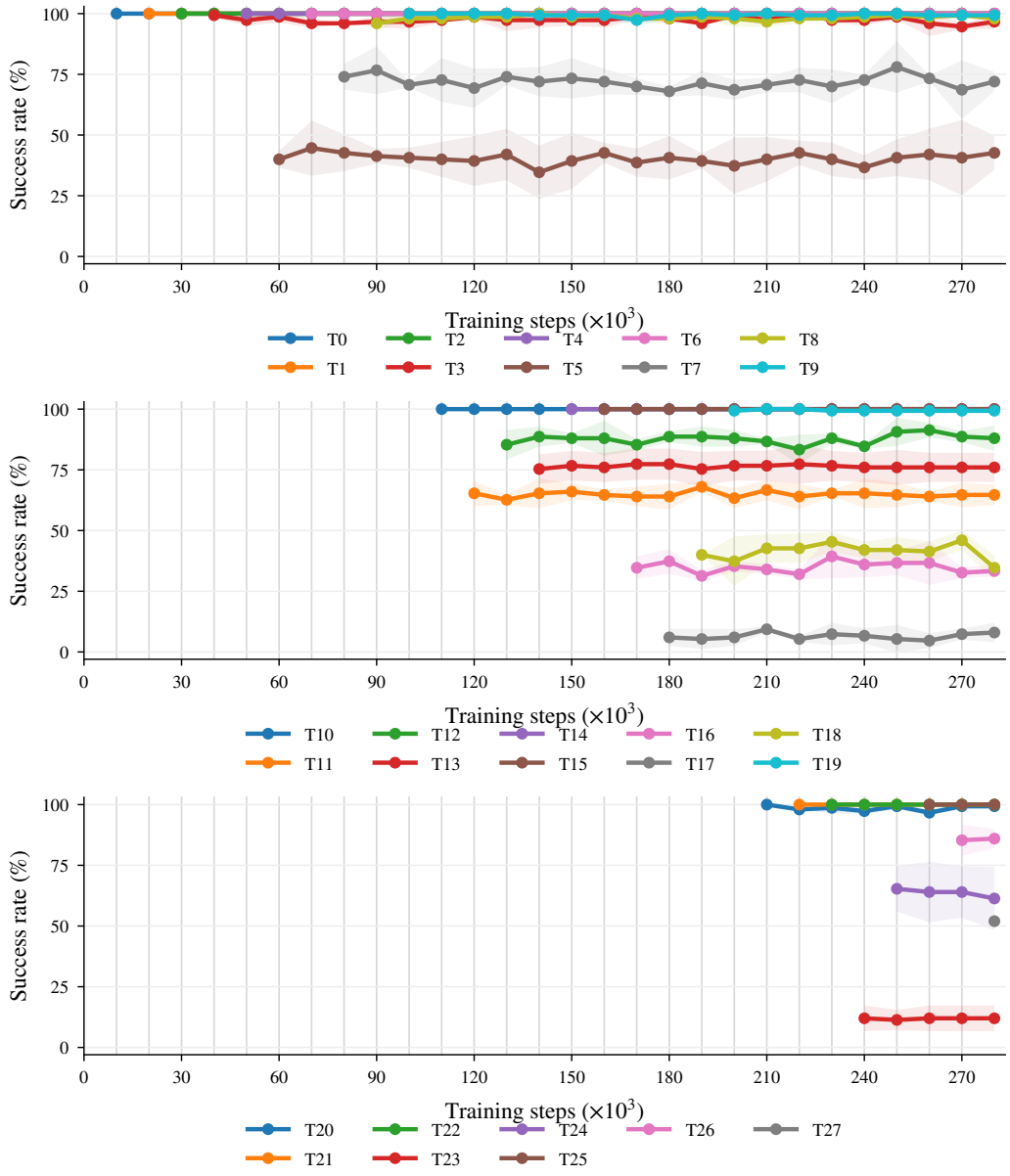


Figure 65: PONIX: task-wise success rates, π_0 on Meta-World Easy. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

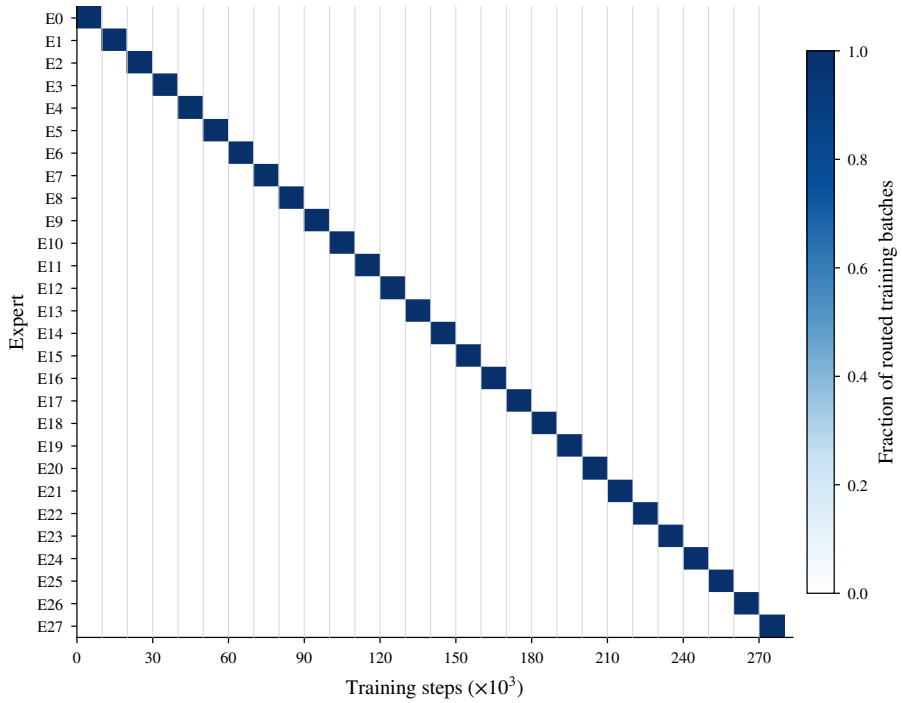


Figure 66: PONIX training activation, π_0 on Meta-World Easy. Fractions of routed batches in 200-update intervals; expert IDs follow creation order.

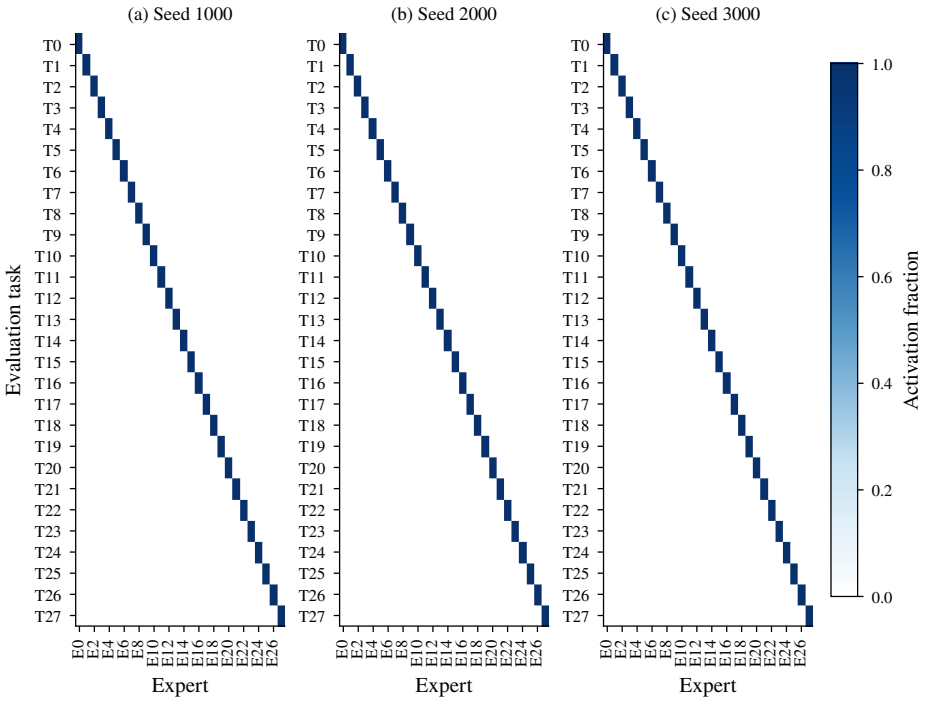


Figure 67: PONIX final evaluation activation, π_0 on Meta-World Easy. Panels correspond to separate evaluation seeds.

B.15 π_0 ON META-WORLD MEDIUM

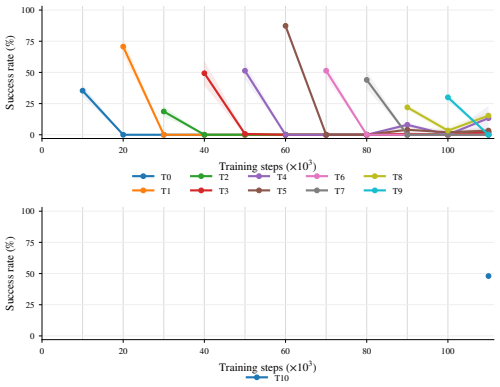


Figure 68: Seq: task-wise success rates, π_0 on Meta-World Medium. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

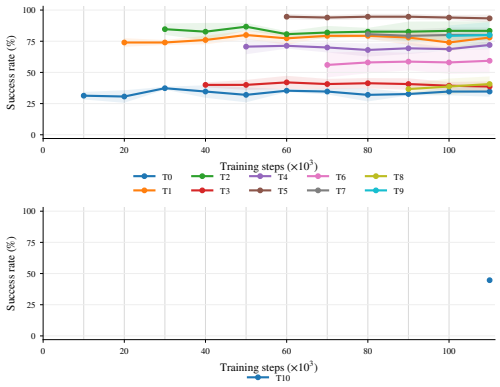


Figure 69: TAIL: task-wise success rates, π_0 on Meta-World Medium. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

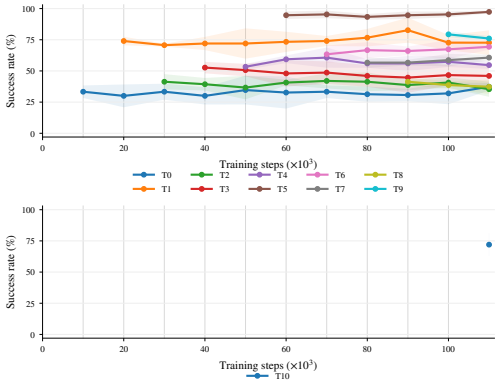


Figure 70: PONIX: task-wise success rates, π_0 on Meta-World Medium. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

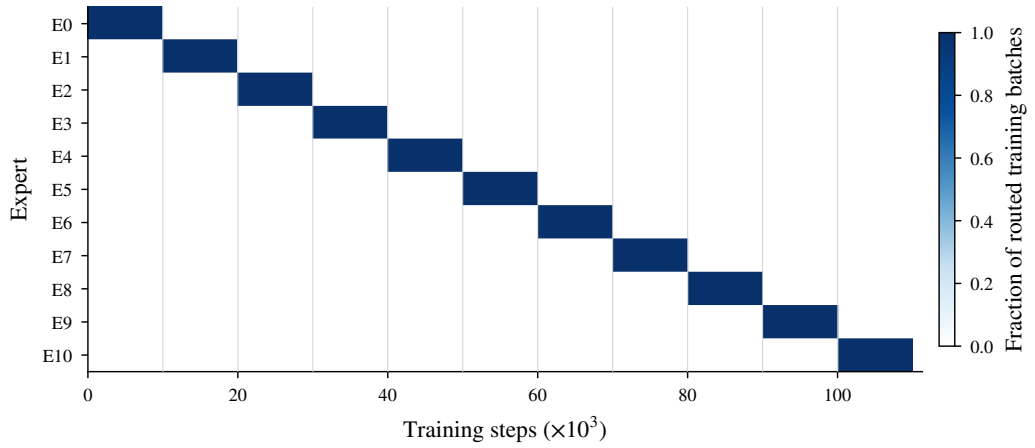


Figure 71: PONIX training activation, π_0 on Meta-World Medium. Fractions of routed batches in 200-update intervals; expert IDs follow creation order.

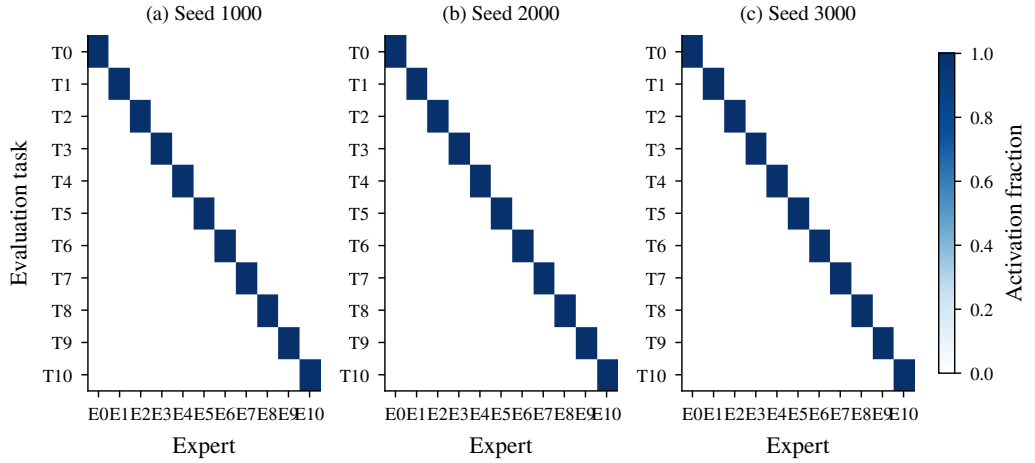


Figure 72: PONIX final evaluation activation, π_0 on Meta-World Medium. Panels correspond to separate evaluation seeds.

B.16 π_0 ON META-WORLD HARD

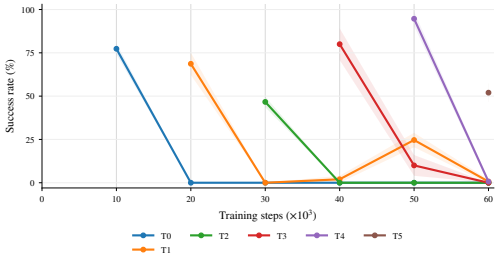


Figure 73: Seq: task-wise success rates, π_0 on Meta-World Hard. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

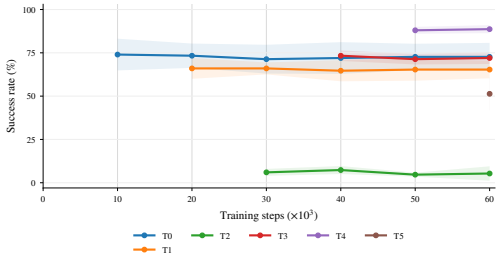


Figure 74: TAIL: task-wise success rates, π_0 on Meta-World Hard. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

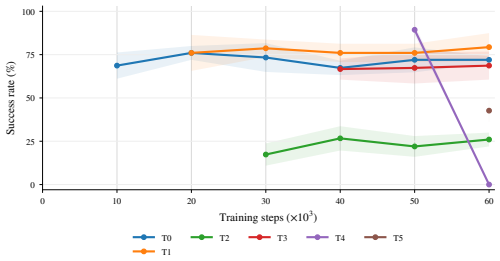


Figure 75: PONIX: task-wise success rates, π_0 on Meta-World Hard. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

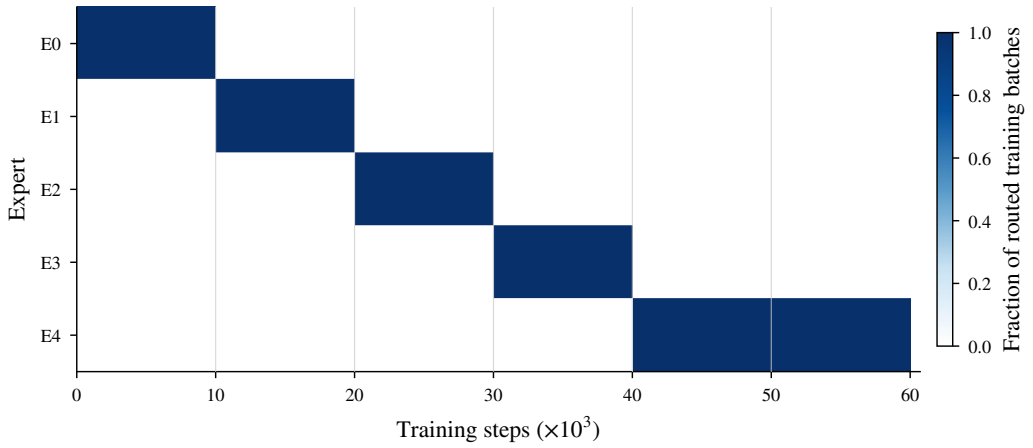


Figure 76: PONIX training activation, π_0 on Meta-World Hard. Fractions of routed batches in 200-update intervals; expert IDs follow creation order.

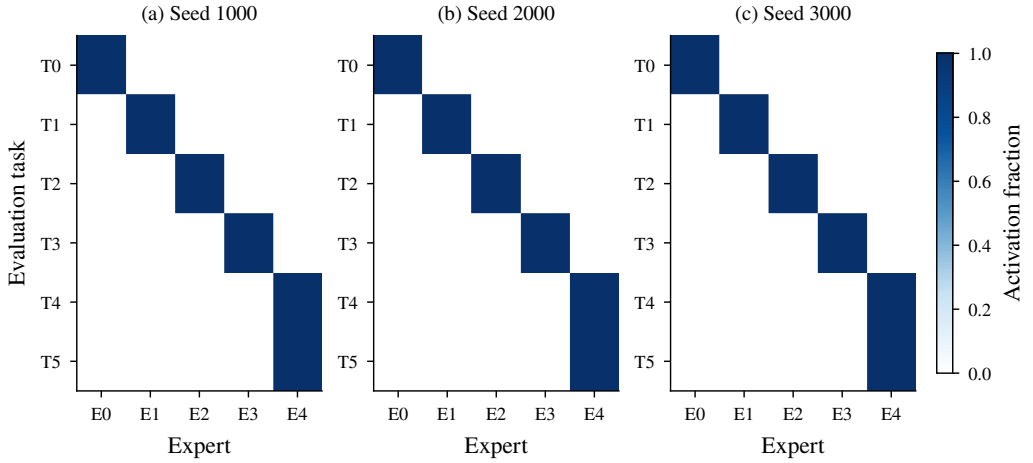


Figure 77: PONIX final evaluation activation, π_0 on Meta-World Hard. Panels correspond to separate evaluation seeds.

B.17 π_0 ON META-WORLD VERY HARD

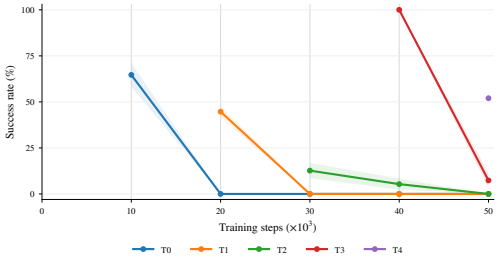


Figure 78: Seq: task-wise success rates, π_0 on Meta-World Very Hard. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

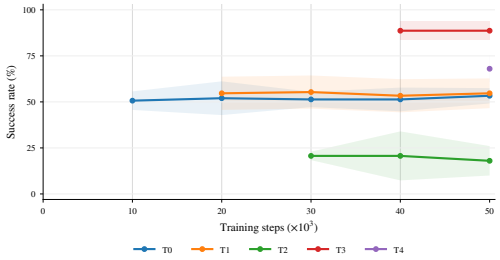


Figure 79: TAIL: task-wise success rates, π_0 on Meta-World Very Hard. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

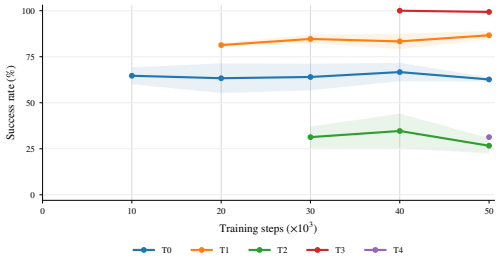


Figure 80: PONIX: task-wise success rates, π_0 on Meta-World Very Hard. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

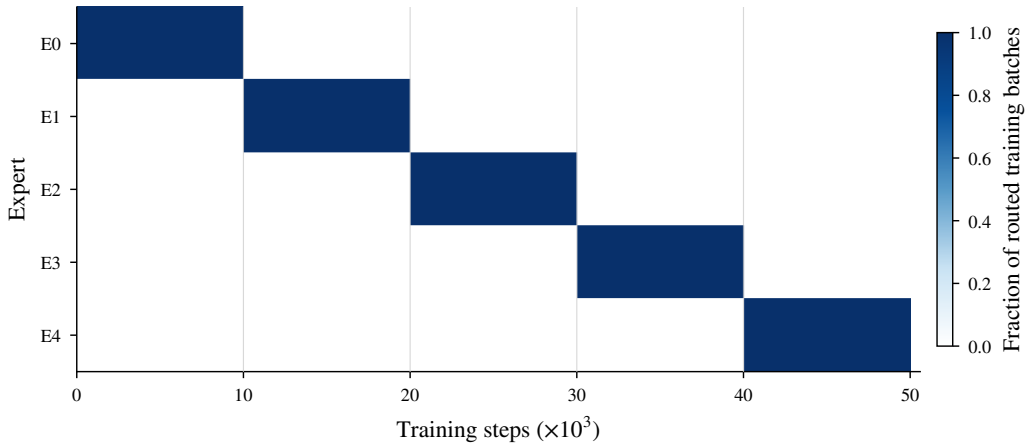


Figure 81: PONIX training activation, π_0 on Meta-World Very Hard. Fractions of routed batches in 200-update intervals; expert IDs follow creation order.

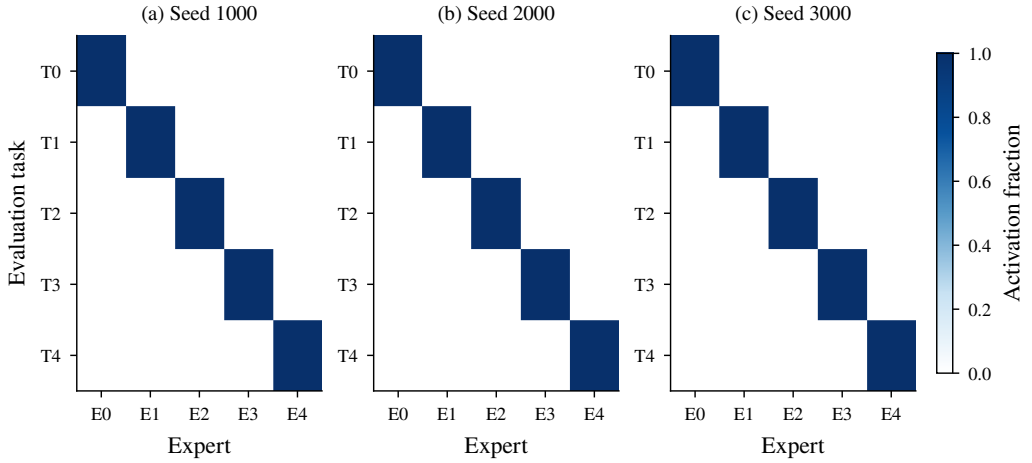


Figure 82: PONIX final evaluation activation, π_0 on Meta-World Very Hard. Panels correspond to separate evaluation seeds.

B.18 GR00T ON LIBERO-SPATIAL

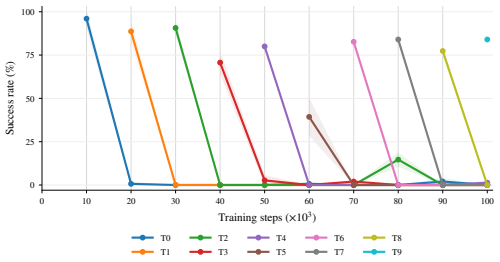


Figure 83: Seq: task-wise success rates, GR00T on LIBERO-Spatial. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

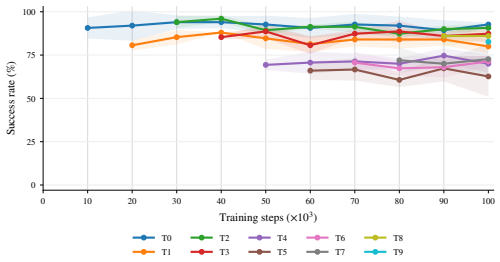


Figure 84: TAIL: task-wise success rates, GR00T on LIBERO-Spatial. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

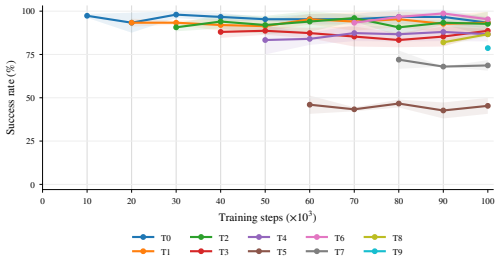


Figure 85: PONIX: task-wise success rates, GR00T on LIBERO-Spatial. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

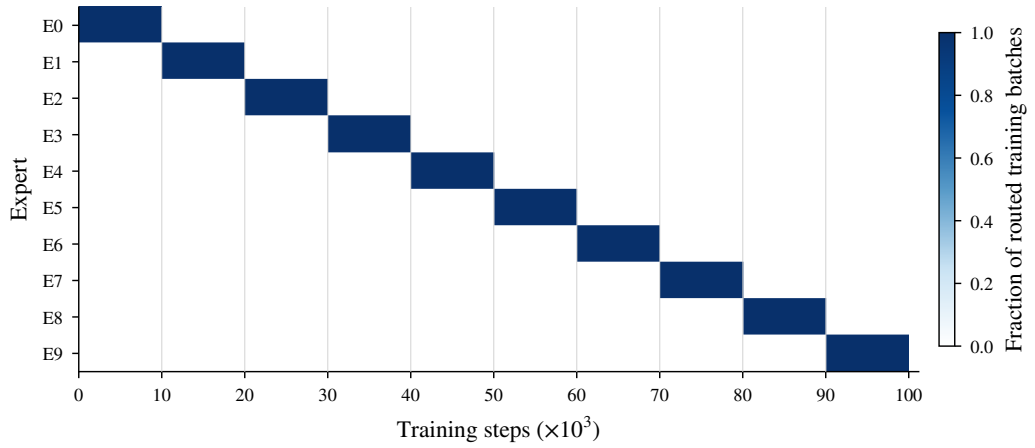


Figure 86: PONIX training activation, GR00T on LIBERO-Spatial. Fractions of routed batches in 200-update intervals; expert IDs follow creation order.

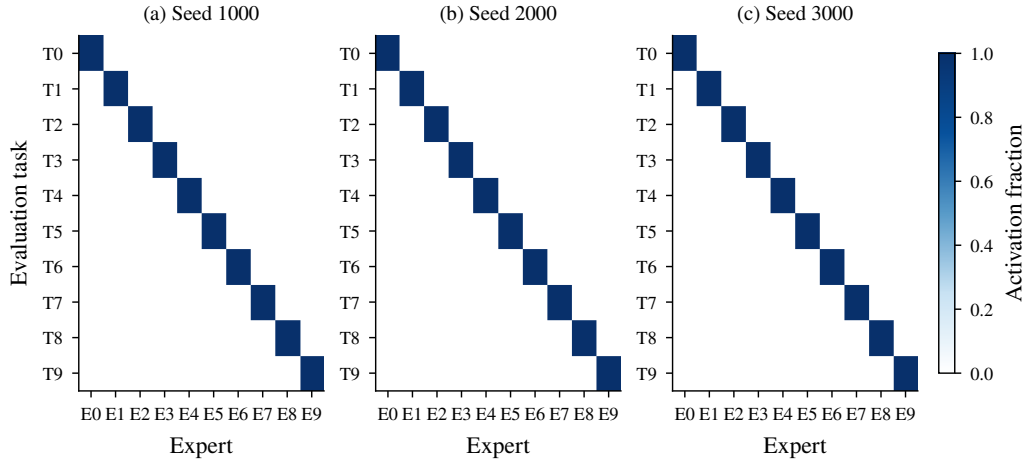


Figure 87: PONIX final evaluation activation, GR00T on LIBERO-Spatial. Panels correspond to separate evaluation seeds.

B.19 GR00T ON LIBERO-OBJECT

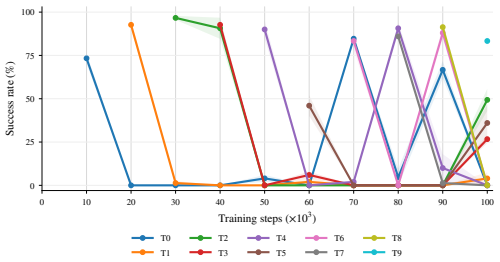


Figure 88: Seq: task-wise success rates, GR00T on LIBERO-Object. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

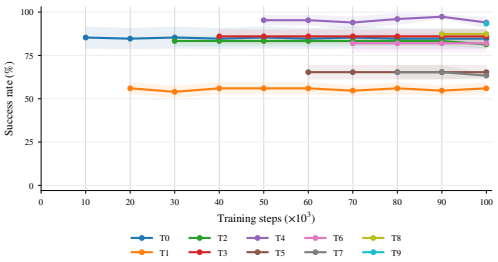


Figure 89: TAIL: task-wise success rates, GR00T on LIBERO-Object. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

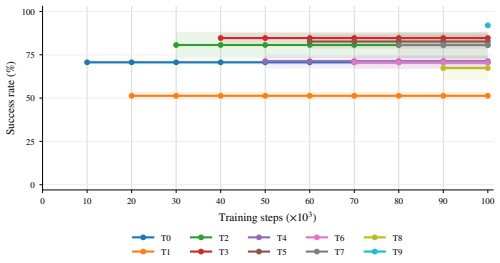


Figure 90: PONIX: task-wise success rates, GR00T on LIBERO-Object. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

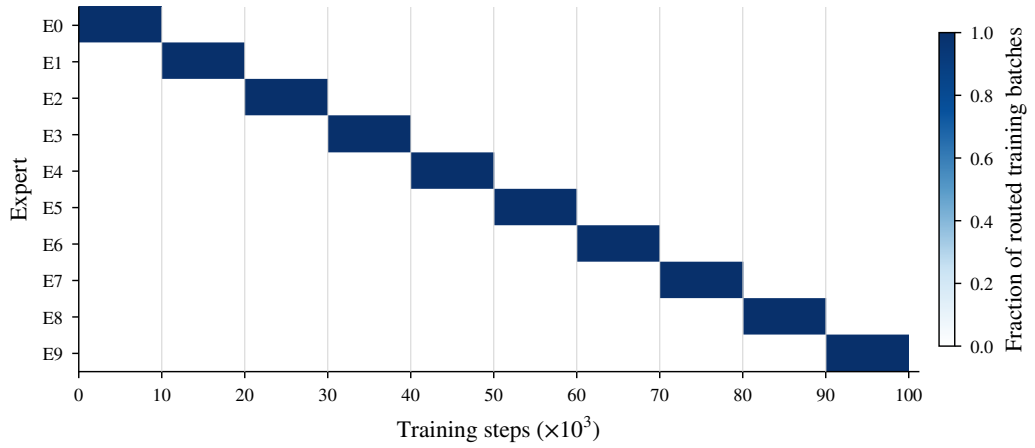


Figure 91: PONIX training activation, GR00T on LIBERO-Object. Fractions of routed batches in 200-update intervals; expert IDs follow creation order.

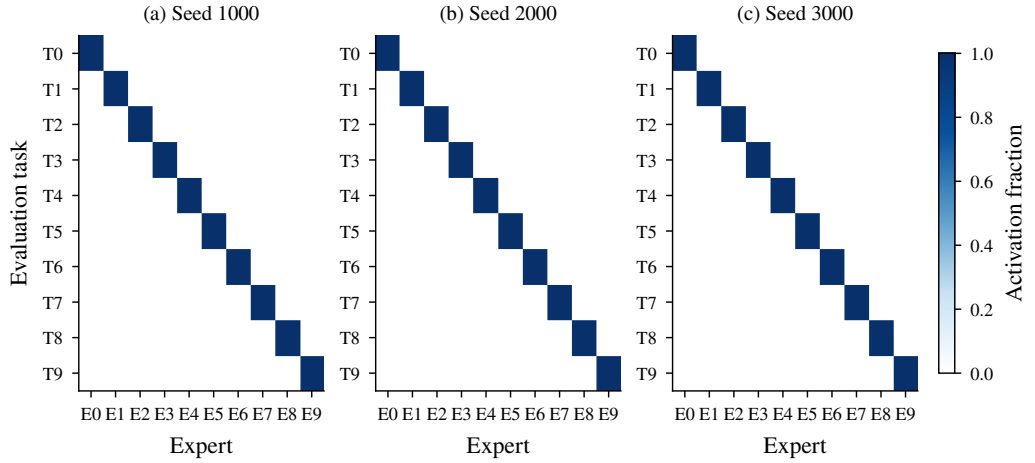


Figure 92: PONIX final evaluation activation, GR00T on LIBERO-Object. Panels correspond to separate evaluation seeds.

B.20 GR00T ON LIBERO-GOAL

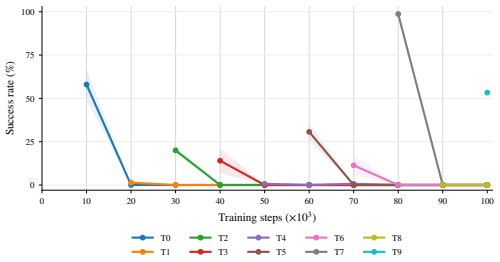


Figure 93: Seq: task-wise success rates, GR00T on LIBERO-Goal. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

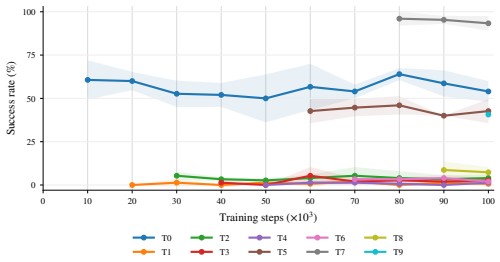


Figure 94: TAIL: task-wise success rates, GR00T on LIBERO-Goal. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

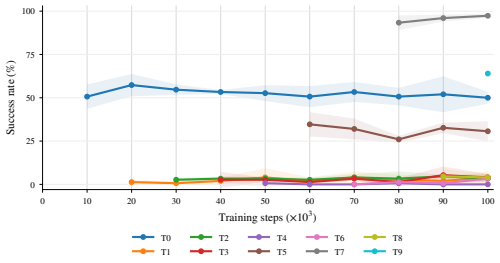


Figure 95: PONIX: task-wise success rates, GR00T on LIBERO-Goal. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

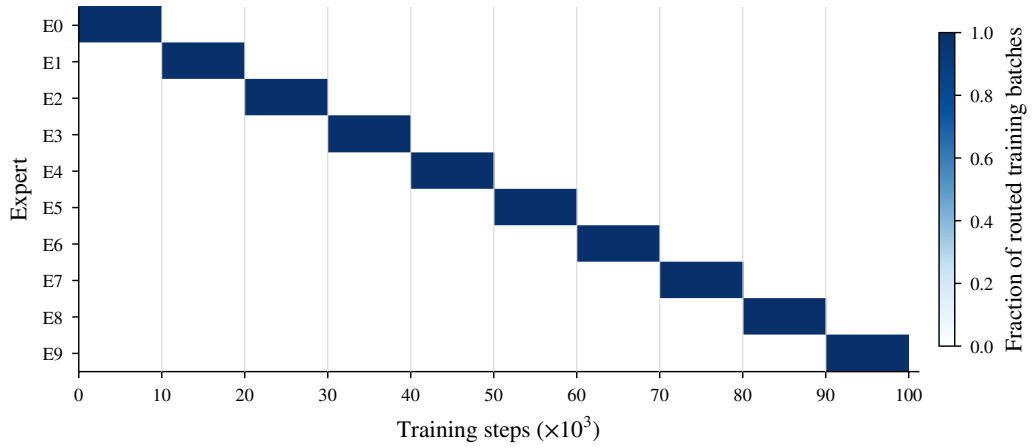


Figure 96: PONIX training activation, GR00T on LIBERO-Goal. Fractions of routed batches in 200-update intervals; expert IDs follow creation order.

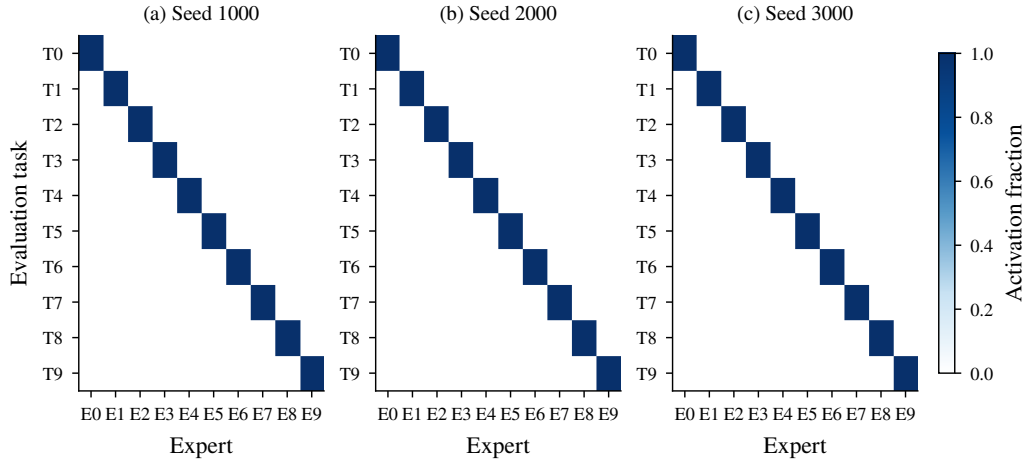


Figure 97: PONIX final evaluation activation, GR00T on LIBERO-Goal. Panels correspond to separate evaluation seeds.

B.21 GR00T ON LIBERO-10

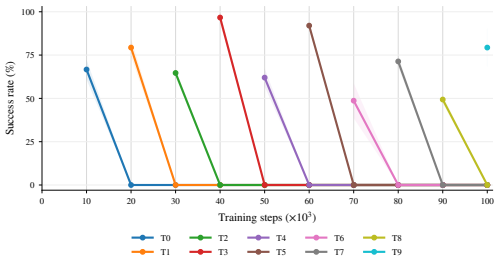


Figure 98: Seq: task-wise success rates, GR00T on LIBERO-10. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

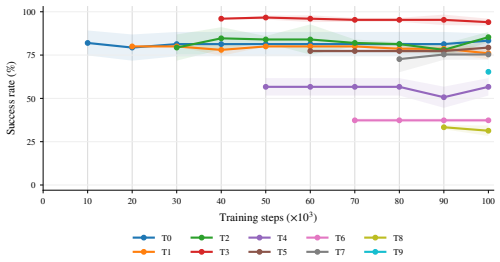


Figure 99: TAIL: task-wise success rates, GR00T on LIBERO-10. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

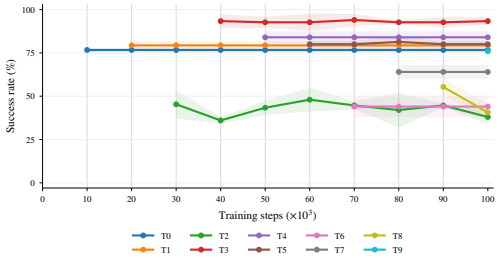


Figure 100: PONIX: task-wise success rates, GR00T on LIBERO-10. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

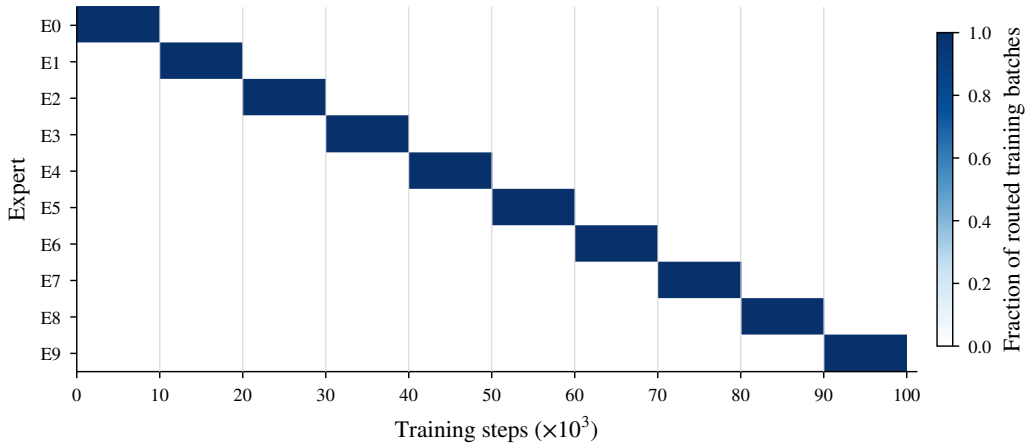


Figure 101: PONIX training activation, GR00T on LIBERO-10. Fractions of routed batches in 200-update intervals; expert IDs follow creation order.

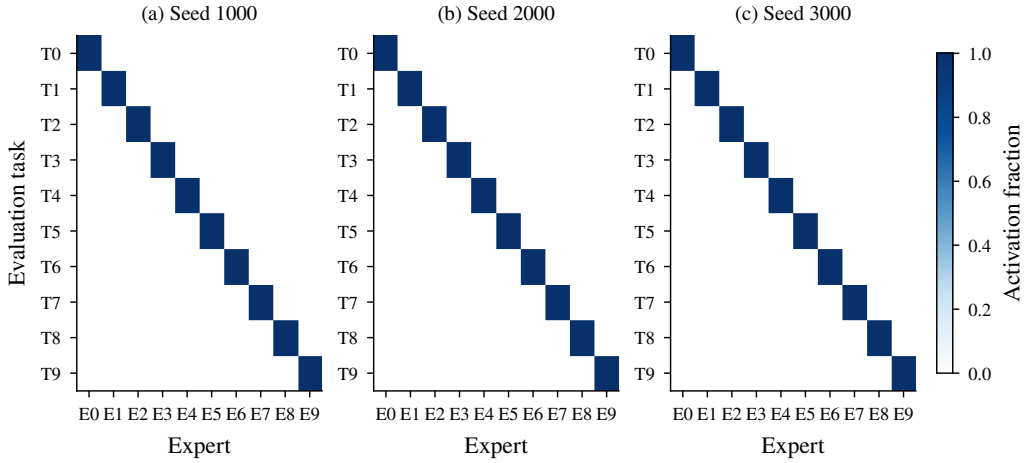


Figure 102: PONIX final evaluation activation, GR00T on LIBERO-10. Panels correspond to separate evaluation seeds.

C TASK-REVISITATION EXPERIMENTS

Each task's eligible episode IDs are sorted and divided into two contiguous, disjoint groups; an odd remainder goes to the first group. Algorithm 2 specifies the stream. Each suite uses 100,000 updates and 23,250 evaluation episodes per method: $3 \times 50(55 + 100)$. On the second pass all ten previously learned tasks are evaluated after every visit.

Algorithm 2 Two-pass revisitation protocol

for each task $k = 0, \ldots, 9$ **do**
    Sort eligible episode IDs D_k; $m_k \leftarrow \lceil |D_k|/2 \rceil$
    $D_k^{(0)} \leftarrow D_k[: m_k]$; $D_k^{(1)} \leftarrow D_k[m_k :]$
for $p = 0, 1$ **do**
    for $k = 0, \ldots, 9$ in the fixed suite order **do**
        Train on $D_k^{(p)}$ for 5,000 updates
        Evaluate every distinct task encountered so far

Table 16: Demonstrations in the first/second visit, ordered T0–T9.

Suite	First/second visit
LIBERO-Spatial	23/22, 23/22, 23/23, 22/21, 21/21, 20/19, 23/23, 18/17, 24/23, 22/22
LIBERO-Object	22/22, 23/22, 24/23, 23/23, 23/22, 21/21, 23/22, 23/22, 25/25, 23/22
LIBERO-Goal	22/21, 24/24, 24/23, 18/18, 23/23, 17/16, 20/20, 25/25, 25/24, 18/18
LIBERO-10	17/16, 25/24, 21/20, 18/17, 19/19, 21/20, 18/18, 22/21, 15/14, 17/17

Table 17: Expert-allocation rules under revisitation. $v \in \{1, \ldots, 20\}$ is the completed visit count. Baseline count curves follow these rules; PONIX counts are measured.

Method	Revisit behavior	Experts after v visits
Seq	Continue updating the same adapter	1
Boundary TAIL	Create a new adapter; evaluate the latest per instruction	v
Instruction TAIL	Reuse the adapter indexed by the instruction	$\min(v, 10)$
PONIX	Infer reuse or expansion from the input batch	Recorded

C.1 SmolVLA on LIBERO-Spatial

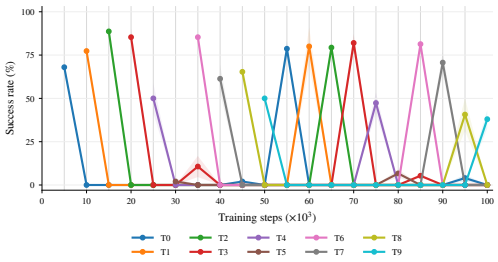


Figure 103: Seq: task-wise success rates, SmolVLA on LIBERO-Spatial. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

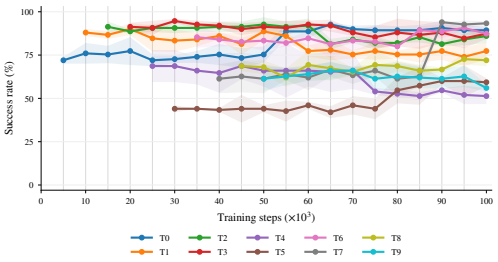


Figure 104: Boundary TAIL: task-wise success rates, SmolVLA on LIBERO-Spatial. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

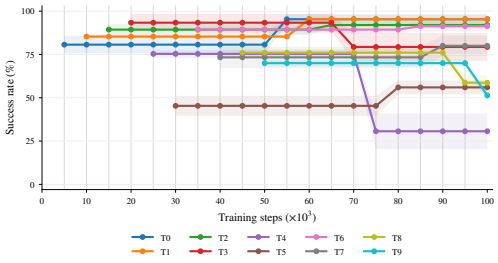


Figure 105: Instruction TAIL: task-wise success rates, SmolVLA on LIBERO-Spatial. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

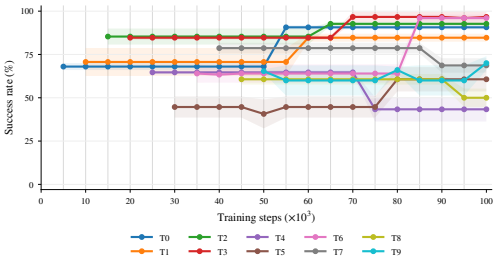


Figure 106: PONIX: task-wise success rates, SmolVLA on LIBERO-Spatial. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

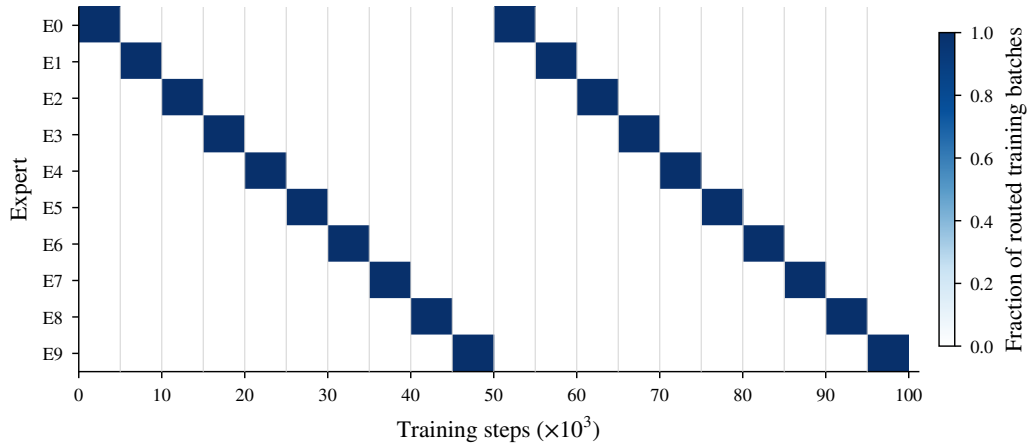


Figure 107: PONIX training activation, SmolVLA on LIBERO-Spatial. Fractions of routed batches in 200-update intervals; expert IDs follow creation order.

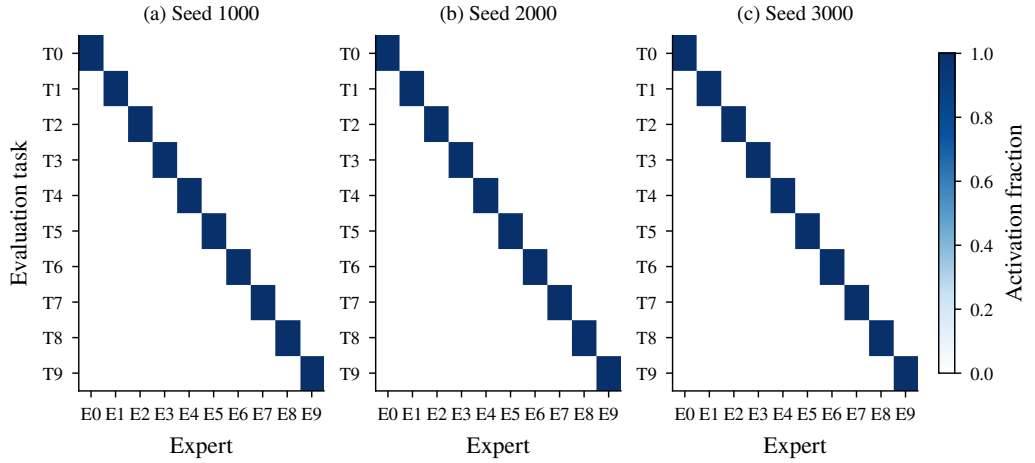


Figure 108: PONIX final evaluation activation, SmolVLA on LIBERO-Spatial. Panels correspond to separate evaluation seeds.

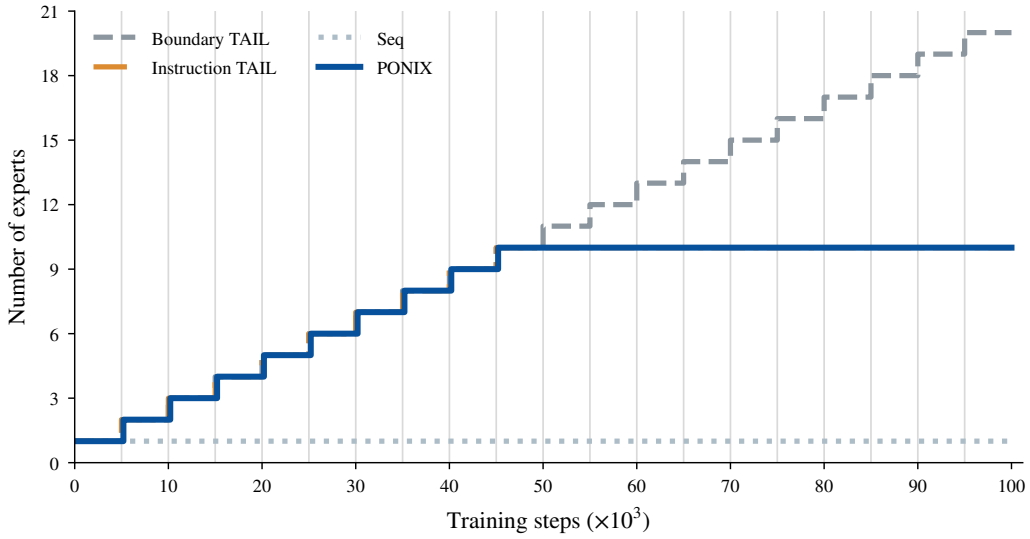


Figure 109: Expert counts during revisitation, SmolVLA on LIBERO-Spatial. PONIX is measured; baseline counts follow their allocation rules.

C.2 SmolVLA on LIBERO-Object

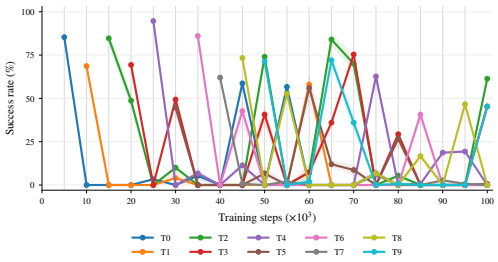


Figure 110: Seq: task-wise success rates, SmolVLA on LIBERO-Object. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

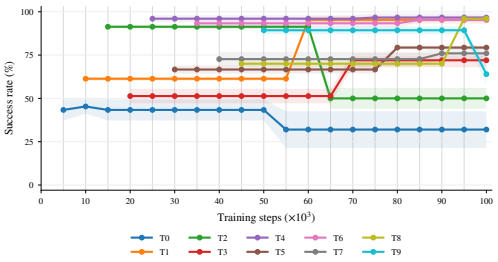


Figure 111: Boundary TAIL: task-wise success rates, SmolVLA on LIBERO-Object. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

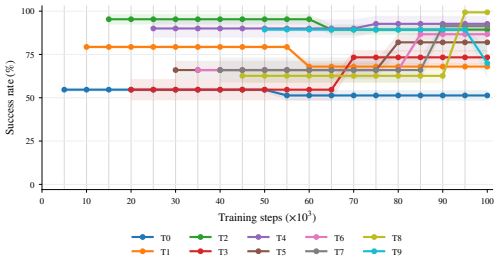


Figure 112: Instruction TAIL: task-wise success rates, SmolVLA on LIBERO-Object. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

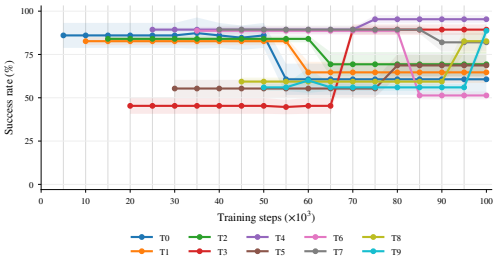


Figure 113: PONIX: task-wise success rates, SmolVLA on LIBERO-Object. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

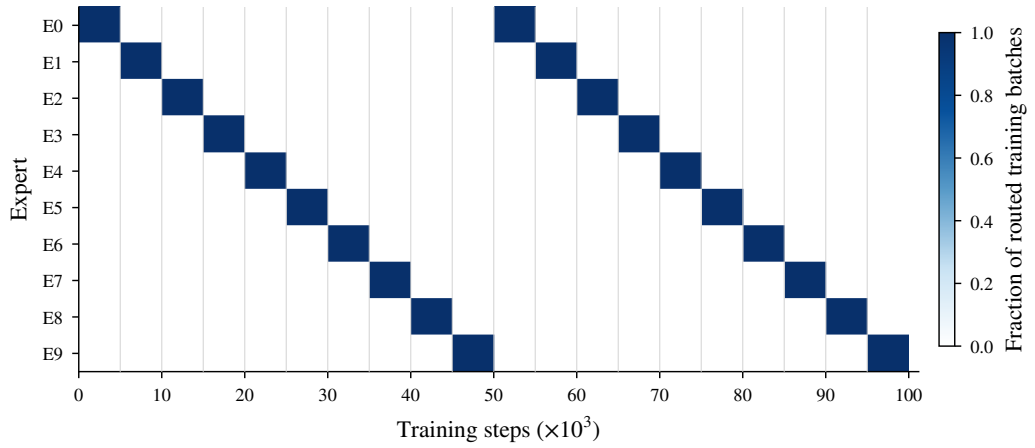


Figure 114: PONIX training activation, SmolVLA on LIBERO-Object. Fractions of routed batches in 200-update intervals; expert IDs follow creation order.

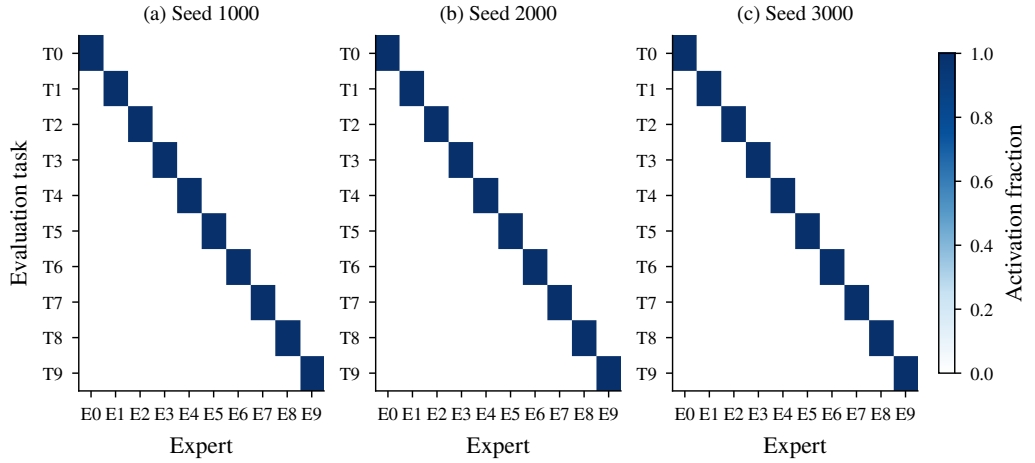


Figure 115: PONIX final evaluation activation, SmolVLA on LIBERO-Object. Panels correspond to separate evaluation seeds.

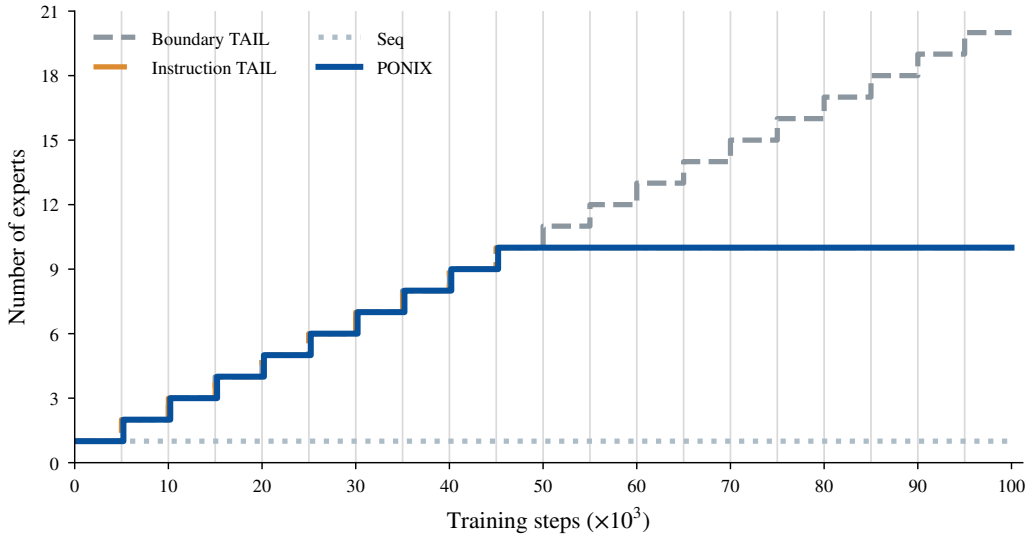


Figure 116: Expert counts during revisitation, SmolVLA on LIBERO-Object. PONIX is measured; baseline counts follow their allocation rules.

C.3 SmolVLA on LIBERO-Goal

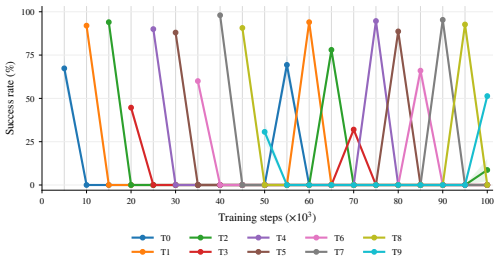


Figure 117: Seq: task-wise success rates, SmolVLA on LIBERO-Goal. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

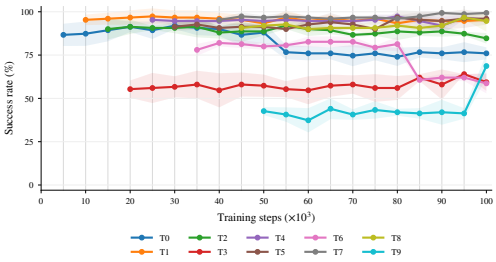


Figure 118: Boundary TAIL: task-wise success rates, SmolVLA on LIBERO-Goal. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

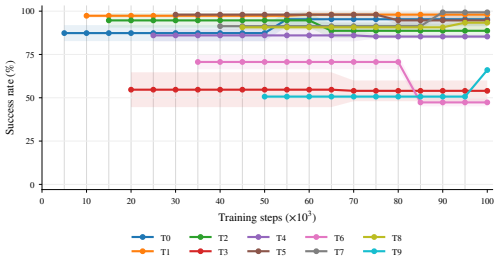


Figure 119: Instruction TAIL: task-wise success rates, SmolVLA on LIBERO-Goal. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

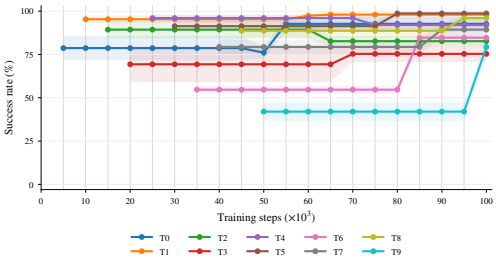


Figure 120: PONIX: task-wise success rates, SmolVLA on LIBERO-Goal. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

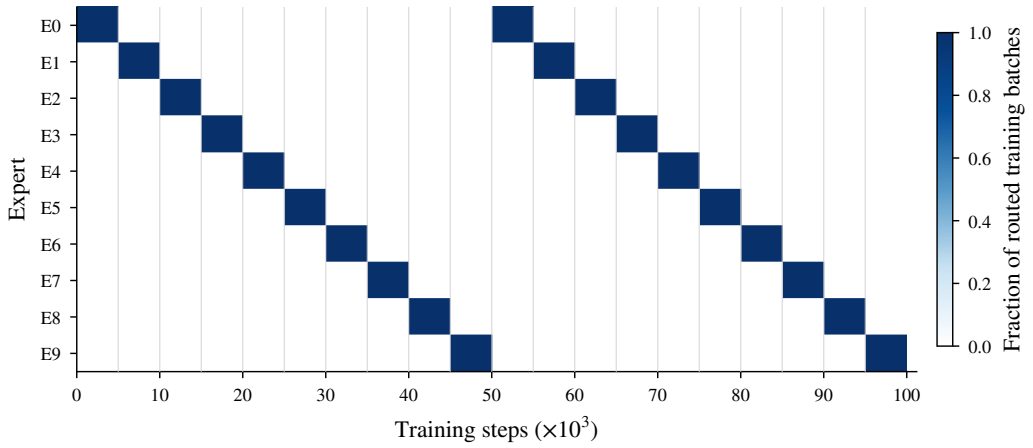


Figure 121: PONIX training activation, SmolVLA on LIBERO-Goal. Fractions of routed batches in 200-update intervals; expert IDs follow creation order.

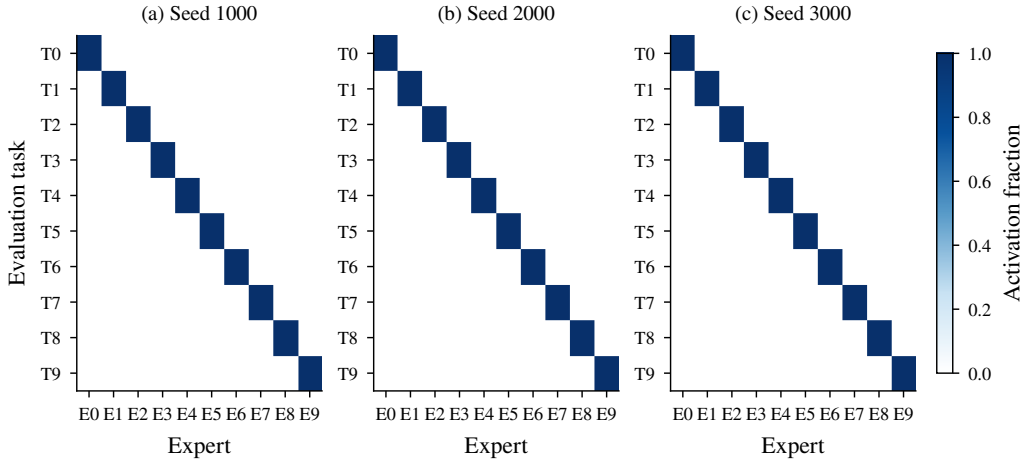


Figure 122: PONIX final evaluation activation, SmolVLA on LIBERO-Goal. Panels correspond to separate evaluation seeds.

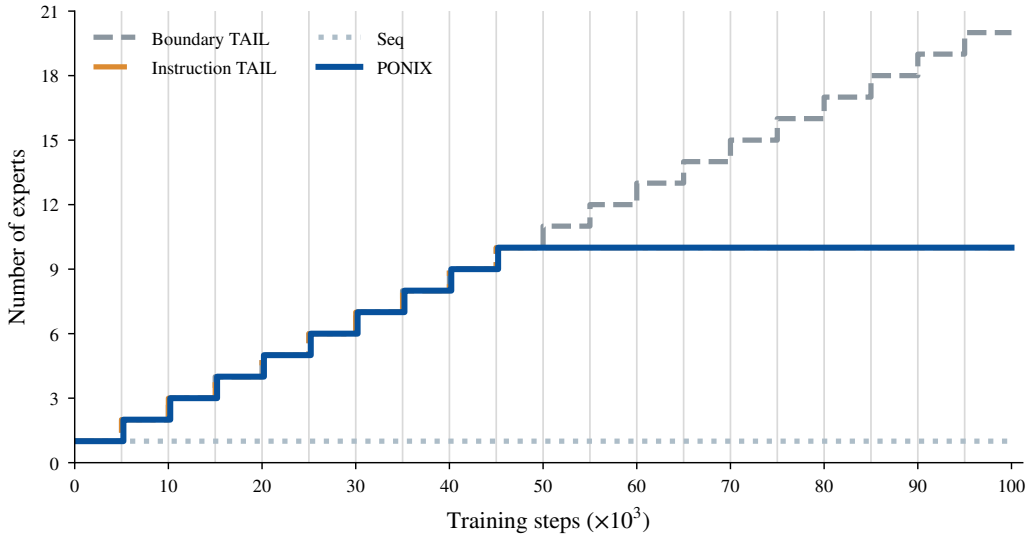


Figure 123: Expert counts during revisitation, SmolVLA on LIBERO-Goal. PONIX is measured; baseline counts follow their allocation rules.

C.4 SmolVLA on LIBERO-10

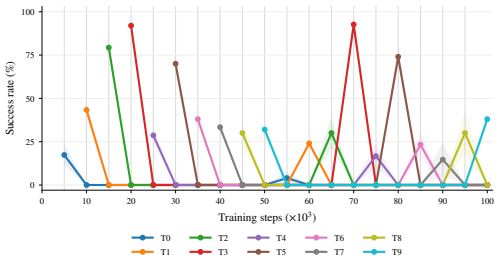


Figure 124: Seq: task-wise success rates, SmolVLA on LIBERO-10. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

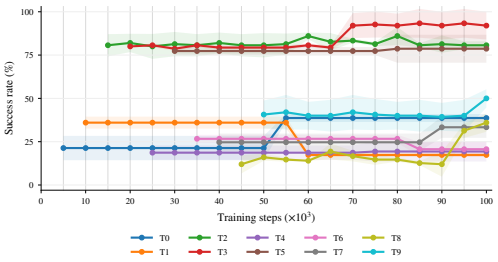


Figure 125: Boundary TAIL: task-wise success rates, SmolVLA on LIBERO-10. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

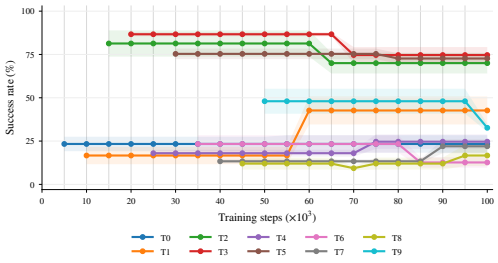


Figure 126: Instruction TAIL: task-wise success rates, SmolVLA on LIBERO-10. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

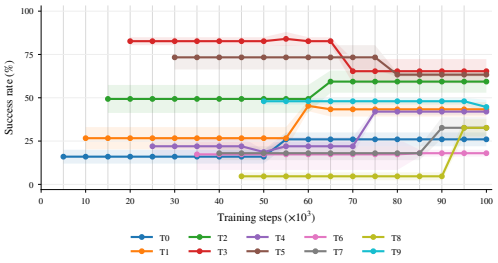


Figure 127: PONIX: task-wise success rates, SmolVLA on LIBERO-10. Mean and sample standard deviation over evaluation seeds 1000, 2000, and 3000.

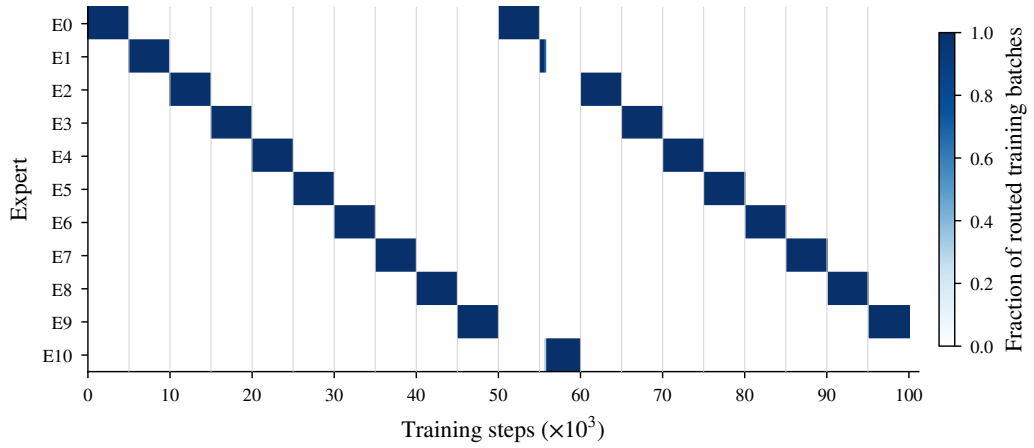


Figure 128: PONIX training activation, SmolVLA on LIBERO-10. Fractions of routed batches in 200-update intervals; expert IDs follow creation order.

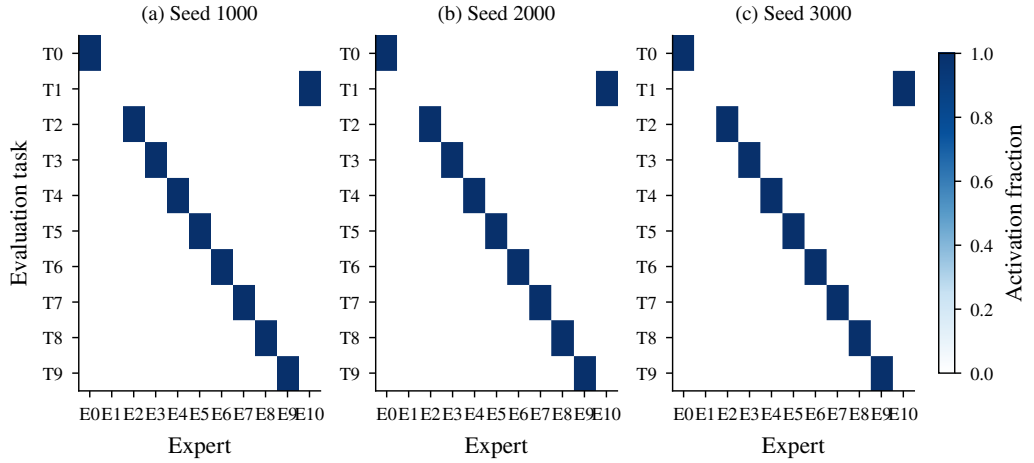


Figure 129: PONIX final evaluation activation, SmolVLA on LIBERO-10. Panels correspond to separate evaluation seeds.

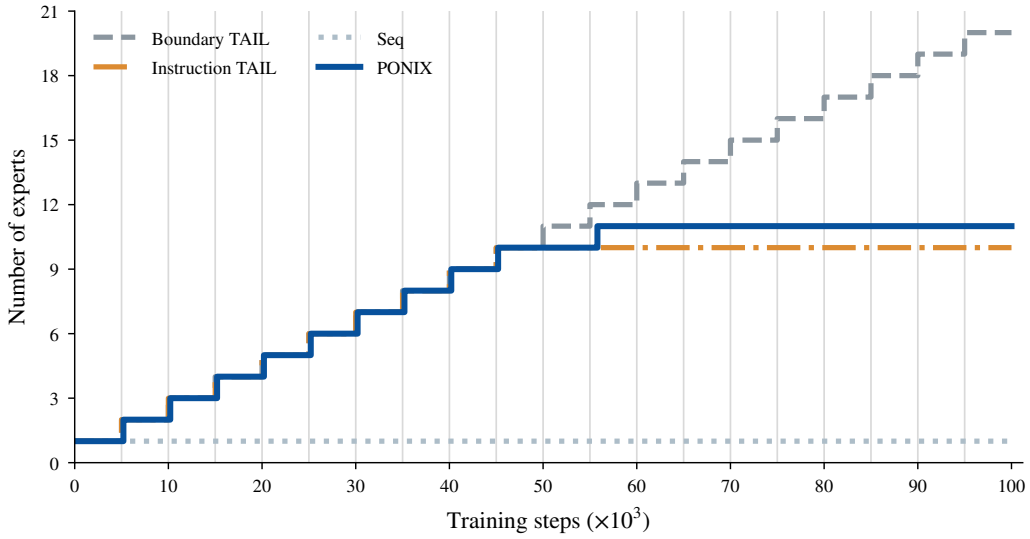


Figure 130: Expert counts during revisitation, SmolVLA on LIBERO-10. PONIX is measured; baseline counts follow their allocation rules.

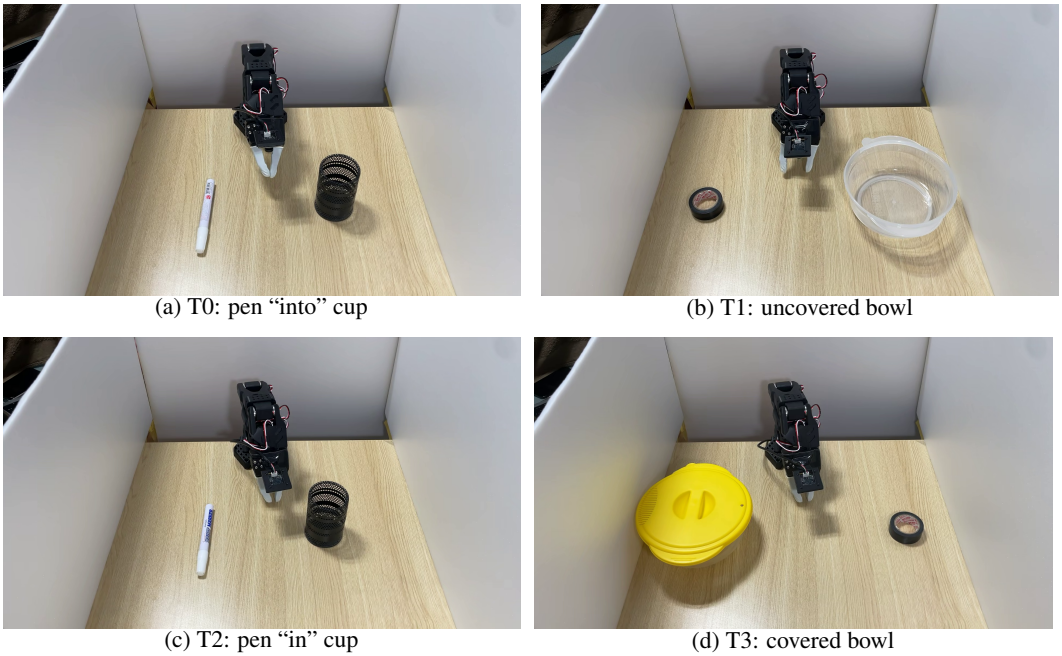
D REAL-ROBOT EXPERIMENTS

Table 18: Real-robot data and execution settings. Client settings were confirmed by the experimenter; training settings come from the saved checkpoint.

Setting	Value
Robot / policy	SO-101 / SmolVLA with PONIX
Dataset / training budget	Our demonstrations; 50 trajectories and 10,000 updates per collection
Cameras	Front and wrist RGB; recorded at 1920×1080, 30 FPS
Policy image preprocessing	Resize with padding to 512×512
State / action	Six joint-position channels, including the gripper
Policy camera mapping	Front $\rightarrow$ camera1; wrist $\rightarrow$ camera2
Routing input	Front-camera token positions plus language, last VLM layer
Control frequency	30 Hz
Actions per chunk / queue threshold	50 / 0.5
Action generation	50-step chunks; 10 denoising steps
Evaluation	Ten finished trials per collection/checkpoint; aborted trials excluded

Table 19: Real-robot stream in training order. Episode IDs are zero-based; 110–119 are excluded. Pen collections T0 and T2 share a task.

Index	Instruction	Initial condition	Episode IDs
T0	Put the pen into the cup	Pen and empty cup	30–39, 170–209
T1	Put the tape in the bowl	Uncovered bowl	20–29, 130–169
T2	Put the pen in the cup	Pen and empty cup	10–19, 80–109, 120–129
T3	Put the tape in the bowl	Covered bowl	0–9, 40–79



(a) T0: pen "into" cup (b) T1: uncovered bowl

(c) T2: pen "in" cup (d) T3: covered bowl

Figure 131: Initial front-camera observations from our demonstration dataset, frame 0 of episodes 30, 20, 10, and 0. These are demonstration initial states, not evaluation outcomes.

Success requires object release inside the container for T0–T2; T3 scores lid opening, not completion of tape placement. All 100 finished trials are matched to routing logs by session ID. Real evaluation activation averages each trajectory's expert-selection fractions with equal weight. One routing event

Table 20: Real-robot successes out of ten finished trials. "—" denotes an unevaluated collection.

Training updates	T0	T1	T2	T3
10,000	6/10	—	—	—
20,000	5/10	3/10	—	—
30,000	8/10	5/10	8/10	—
40,000	9/10	$4/10^{\dagger}$	7/10	3/10

† The supplied evaluation directory assigns these trials to 40k, but their server logs identify 30k. The SR curve flags this point; activation is displayed at the supplied 40k assignment. It is not verified final-checkpoint retention.

denotes a generated chunk, not necessarily 50 executed actions. Front-only token pooling retains the joint two-camera VLM forward pass, so the selected features can still contain contextual information from the wrist view.

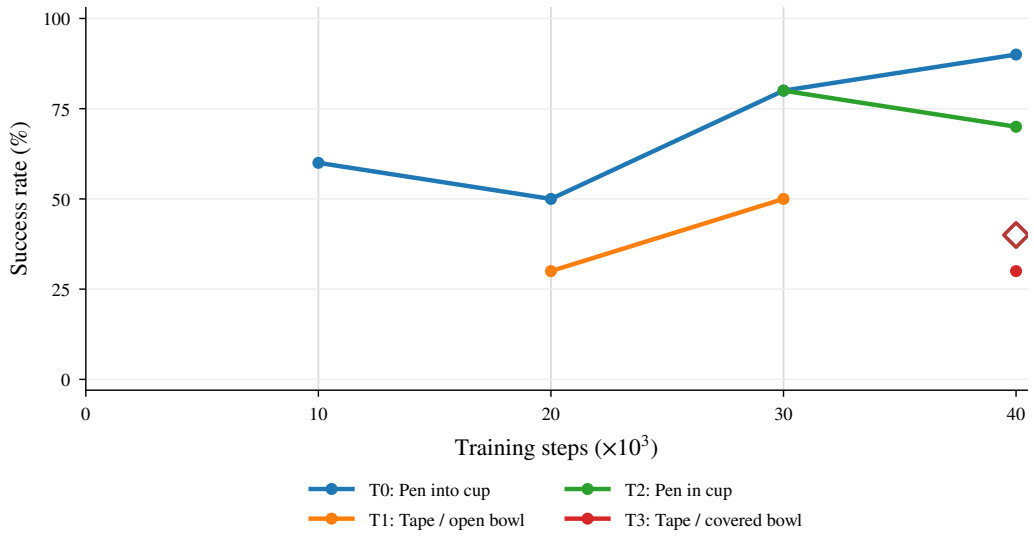


Figure 132: PONIX: task-wise success rates, SmolVLA on Real robot. Ten finished trials per point; the hollow diamond denotes the checkpoint discrepancy in Table 20.

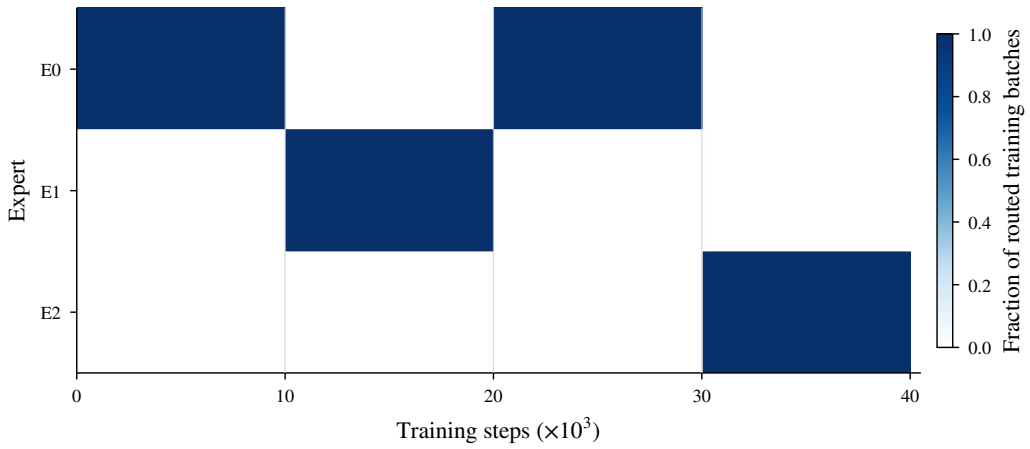


Figure 133: PONIX training activation, SmolVLA on Real robot. Fractions of routed batches in 200-update intervals; expert IDs follow creation order.

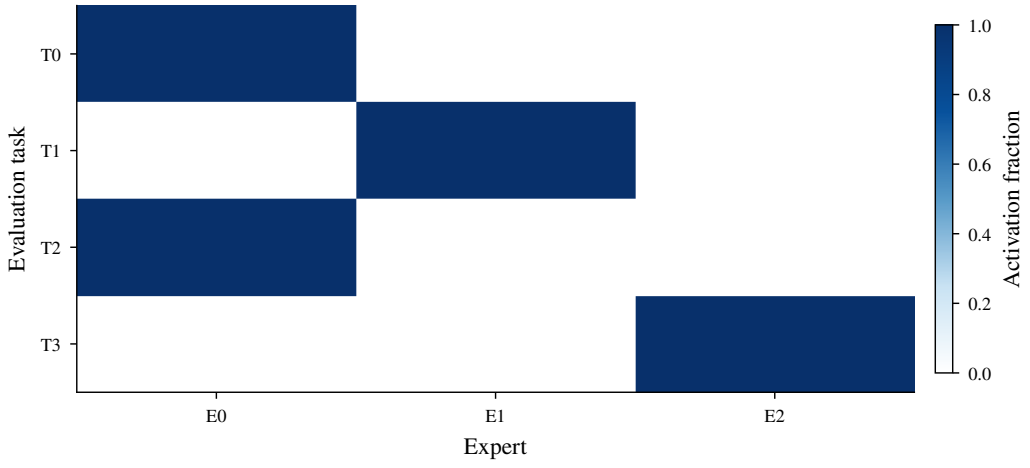


Figure 134: PONIX final evaluation activation, SmolVLA on Real robot. Trajectory-normalized chunk-selection fractions; T1 follows the supplied 40k assignment (Table 20).

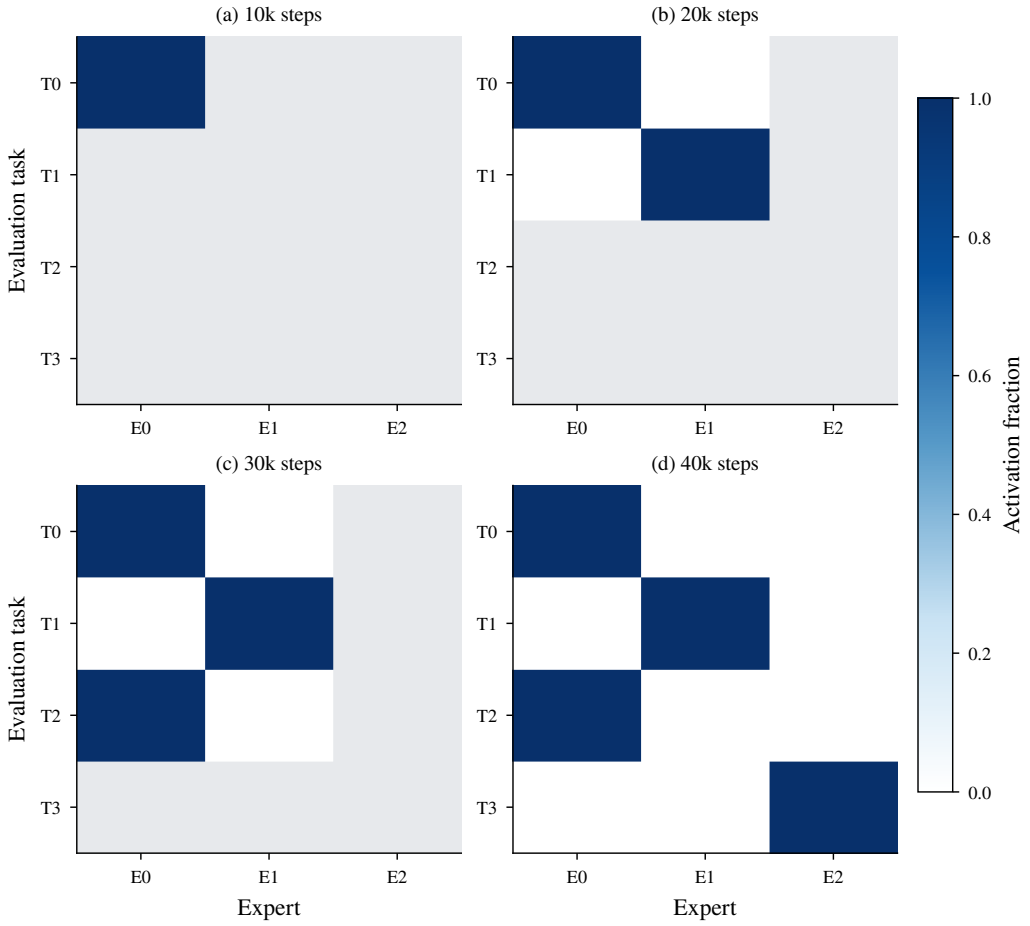


Figure 135: Real-robot evaluation activation at all four checkpoints. T1 at 40k follows the supplied assignment; its server log retains a 30k checkpoint field.