POLICY-SOUP: GENERALIZABLE MANIPULATION THROUGH HETEROGENEOUS EXPERT MERGING

Anonymous authors
Paper under double-blind review

ABSTRACT

Robust visuomotor manipulation requires policies to operate across diverse domains. Domain-specialized experts provide a natural way to handle domain variation, but maintaining separate experts does not provide a unified solution for mixed or previously unseen combinations of domain variations. An important yet underexplored question is therefore how to consolidate their complementary capabilities into a single reusable policy. This becomes particularly challenging when experts differ in model capacity, as conventional parameter merging assumes structural compatibility. Thus, we propose **POLICY-SOUP**, a method for consolidating heterogeneous policy experts into a unified policy. Our key insight is that different policy components encode different degrees of transferable and specialized knowledge. POLICY-SOUP shares transferable information through learned fusion of expert visual features and layer-wise aggregation of attention outputs, while preserving specialization by retaining heterogeneous feed-forward networks at their original widths and selecting among them through sequence-level sparse routing. The unified policy is trained on pooled source-domain demonstrations, without requiring data for every combination of environmental variations. We evaluate POLICY-SOUP on RoboTwin 2.0, LIBERO, PushT, and real-world manipulation across cross-environment, cross-task, and cross-embodiment settings. POLICY-SOUP preserves strong source-domain performance, improves robustness to held-out and compositional shifts, and consolidates capabilities across tasks and robot embodiments.

1 INTRODUCTION

Diffusion policies Chi et al. (2024) have emerged as a powerful approach to visuomotor learning. However, a policy that performs well in a particular environment can still be highly sensitive to changes in its visual and environmental conditions. Variations in lighting, background texture, or object appearance may induce substantial distribution shifts, and the challenge becomes even more pronounced when multiple variations occur simultaneously. A natural approach is to train domain-specialized experts, allowing individual policies to capture the characteristics of different environments. Yet maintaining a collection of specialized policies introduces additional deployment complexity and, more importantly, does not provide a unified solution for mixed or previously unseen combinations of domain variations. This raises a fundamental yet comparatively underexplored problem in robotic manipulation: *how to consolidate the complementary knowledge of multiple domain-specialized policies into a single policy without sacrificing their specialized capabilities?*

Model merging Fu et al. (2026); Yadav et al. (2026) provides a natural way to consolidate multiple specialized policies into a single model. However, these methods are primarily designed for structurally homogeneous models and are not well suited to heterogeneous policies. In robotic manipulation, experts trained on different domains or tasks may require different model capacities due to varying environmental complexity and task difficulty. Such heterogeneous experts cannot be directly consolidated through conventional parameter merging. Mixture-of-Experts (MoE) provides a more flexible alternative by routing inputs to different specialized experts Shazeer et al. (2017); Jacobs et al. (1991). However, existing MoE policies Kuzmenko & Shvai (2026); Hao et al. (2026); Celik et al. (2022); Huang et al. (2025) also assume homogeneous experts with identical architectures and capacities, and thus are not well suited to heterogeneous policies.

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.

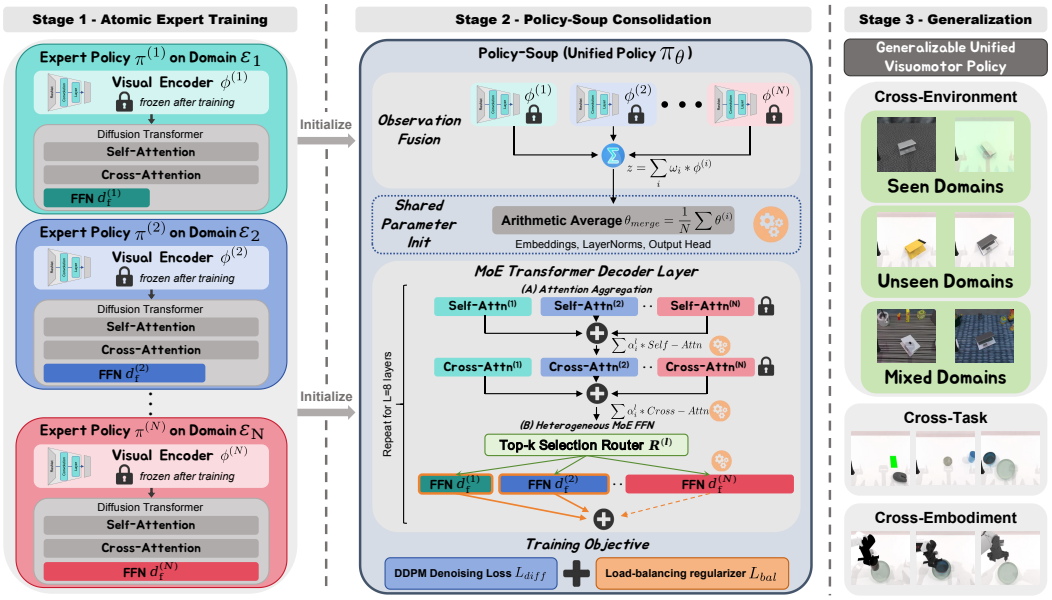

Figure 1: **Overview of POLICY-SOUP.** POLICY-SOUP consolidates heterogeneous diffusion-policy experts into a unified visuomotor policy. Stage 1: Specialized experts are independently trained on source domains, with potentially different FFN widths. Stage 2: POLICY-SOUP consolidates these experts by sharing transferable knowledge through learnable fusion of frozen encoder features and layer-wise aggregation of frozen attention outputs, while preserving specialization via sequence-level routing over heterogeneous FFN experts. Stage 3: The resulting unified policy is evaluated directly on seen, unseen, and mixed domains across environments, tasks and embodiments without additional adaptation.

To address this challenge, we propose POLICY-SOUP, a trainable framework that extends the MoE paradigm to consolidate heterogeneous, pretrained diffusion-policy experts into a unified visuomotor policy. Our key insight is that the knowledge encoded in different components of an expert policy has different degrees of domain specificity. In particular, we retain the feed-forward networks (FFNs) as specialized experts whose capacities are chosen to reflect differences in manipulation difficulty across environments, tasks, and robot embodiments. Across all three consolidation settings, each expert's FFN hidden dimension is determined before expert training through preliminary difficulty assessments. Rather than directly merging their incompatible parameters, we preserve the pretrained FFNs at their original widths and use a sequence-level top-k router to dynamically select and combine their outputs. In this way, POLICY-SOUP accommodates experts with different FFN hidden dimensions while requiring compatible layer interfaces.

However, retaining heterogeneous FFN experts alone is insufficient, as the unified policy must also share transferable knowledge across domains. We therefore learn to fuse the features produced by expert visual encoders and aggregate their attention outputs with layer-wise weights, allowing the policy to share common representations while preserving domain-specific knowledge in the heterogeneous FFN experts. The expert encoders and attention modules remain frozen to preserve their pretrained representations, while compatible shared components are initialized by averaging their corresponding parameters. We then jointly optimize the fusion and aggregation weights, MoE router, FFN experts, and shared components using pooled demonstrations from the atomic domains.

We evaluate POLICY-SOUP across three complementary settings: cross-environment, cross-task, and cross-embodiment consolidation. The cross-environment experiments span RoboTwin 2.0 Chen et al. (2025), LIBERO Liu et al. (2023), and PushT Chi et al. (2024), and cover seen atomic domains, held-out domains, and mixed domains combining multiple variations. Across these evaluations, POLICY-SOUP maintains strong seen-domain performance and improves on individual experts under several held-out and compositional shifts. Separate experiments on RoboTwin 2.0 examine cross-task and cross-embodiment consolidation, while two real-world experiments assess the framework in physical manipulation settings.

Our contributions are as follows:

- We investigate the underexplored problem of consolidating multiple domain-specialized policy experts into a single policy, an important step toward robust and scalable robotic manipulation.
- We propose POLICY-SOUP, a framework that extends the MoE paradigm to heterogeneous experts. It preserves domain-specific knowledge by retaining FFN experts with difficulty-guided widths and sequence-level routing, while sharing transferable knowledge through learnable feature fusion and layer-wise attention aggregation.
- We evaluate POLICY-SOUP across cross-environment, task, and embodiment settings on RoboTwin 2.0, LIBERO, PushT, and real-world platforms. Results show that POLICY-SOUP preserves in-domain accuracy while enhancing generalization to held-out and compositional shifts.

2 RELATED WORK

Diffusion Policies. Diffusion policies generate actions through iterative denoising to model multimodal visuomotor behaviours Chi et al. (2024). Advances in imitation learning Jang et al. (2022); Fu et al. (2024) complement generalist policies trained on diverse robot data Brohan et al. (2022; 2023); Team et al. (2024); Kim et al. (2024); Bousmalis et al. (2023) and language-grounded perception and action Driess et al. (2023); Ahn et al. (2022). Despite this progress, generalization beyond training conditions remains challenging Gao et al. (2026); Xiang et al. (2026); Zhu et al. (2025). Our focus is on consolidating capabilities from specialized diffusion policies rather than learning them jointly from the outset.

Model Parameter Merging. Weight averaging and interpolation combine compatible models without retaining separate inference branches Izmailov et al. (2019); Wortsman et al. (2022a;b), while task-vector and interference-aware methods combine task-specific updates and address conflicts Ilharco et al. (2023); Yadav et al. (2023b); Yu et al. (2024); Huang et al. (2024); Cheng et al. (2025). Robotic applications address cross-task integration and robust adaptation of vision-language-action policies Fu et al. (2026); Yadav et al. (2026). Direct parameter merging requires matching tensor shapes, precluding direct averaging of FFNs with different hidden widths. POLICY-SOUP retains these FFNs with compatible layer interfaces and uses parameter averaging only to initialize compatible shared components.

Mixture-of-Experts. MoE architectures combine specialized modules through learned gating Jacobs et al. (1991), with sparse routing enabling conditional computation Shazeer et al. (2017); Fedus et al. (2022); Du et al. (2022); Jiang et al. (2024). Robotic applications include skill specialization, multi-task policies, and visual reinforcement learning Kuzmenko & Shvai (2026); Hao et al. (2026); Celik et al. (2022); Huang et al. (2025); Model MoErging more broadly studies expert reuse Yadav et al. (2025). Closely related, Branch-Train-MiX assembles independently trained language-model FFNs into MoE layers, averages remaining parameters, and learns token-level routing Sukhbaatar et al. (2024). POLICY-SOUP shares this expert-reuse principle but consolidates diffusion policies with heterogeneous FFN widths, sequence-level routing, and learned layer-wise aggregation of frozen attention outputs.

3 METHODOLOGY

POLICY-SOUP consolidates diffusion-policy experts specialized in different environments, tasks, or embodiments while retaining their heterogeneous feed-forward network (FFN) widths. Figure 1 illustrates expert training (Stage 1), policy consolidation (Stage 2), and evaluation (Stage 3) for the cross-environment, cross-task and cross-embodiment settings. Only the first two stages involve optimization; the same consolidation architecture supports all three settings.

3.1 PROBLEM FORMULATION

We consider N domain-specialized policies $\{\pi_i\}_{i=1}^{N}$, each trained on demonstrations from a source domain $\mathcal{D}_i$. The meaning of a source domain depends on the consolidation experiment. For cross-environment consolidation, $\mathcal{D}_i$ is an atomic environmental domain defined by a controlled category of variation, such as lighting, texture, object appearance, or sensor noise, while the manipulation task

and robot embodiment remain fixed. For cross-task consolidation, experts specialize in different tasks on the same robot embodiment. For cross-embodiment consolidation, experts specialize in the same task on different robot embodiments. Let $\mathcal{B}_i = \{(\mathbf{o}, \mathbf{a})\}$ denote the demonstration set associated with domain $\mathcal{D}_i$, where $\mathbf{o}$ is an observation history and $\mathbf{a}$ is the action sequence.

At deployment, multiple environmental factors may vary simultaneously, resulting in previously unseen compositions of atomic domains. Such mixed conditions can induce distribution shifts that are not covered by any individual expert. Training and maintaining a separate policy for every possible combination is impractical and does not provide a unified mechanism for sharing complementary knowledge across experts. Our objective is therefore to consolidate these specialized experts into a single visuomotor policy,

$$\pi_{\text{uni}} = \mathcal{C}\left(\{\pi_i, \mathcal{B}_i\}_{i=1}^{N}\right), \tag{1}$$

where $\mathcal{C}$ denotes the consolidation procedure. The unified policy should preserve source-domain capabilities while leveraging complementary expert knowledge to generalize to held-out and unseen domain compositions, despite structural heterogeneity among source experts.

3.2 Heterogeneous Policy Consolidation

We now describe how the pretrained experts are consolidated into a single policy. The main challenge is that different components admit different forms of reuse: some parameters are structurally compatible and can be shared directly, while the heterogeneous FFNs cannot be merged parameter-wise due to their different hidden widths. We therefore integrate expert representations at the observation, attention, and FFN levels within a single forward computation, and jointly optimize the trainable components during consolidation.

We first initialize all compatible shared components, including the action and positional embeddings, observation-conditioning network, layer normalization, and noise-prediction head, by averaging the corresponding expert parameters. In contrast, the pretrained visual encoders, attention modules, and FFNs are retained as separate expert components. The encoders and attention modules remain frozen during consolidation, while the FFNs remain trainable and preserve their original hidden widths. The unified policy is then optimized on the pooled demonstrations from all source experts. For clarity, we describe a single decoder layer below and omit layer indices.

Observation fusion. Since each encoder is trained on a different source domain, its features capture complementary visual cues. We combine frozen expert-encoder features using a learned weighted average. Softmax-normalized weights are shared across inputs rather than predicted separately for each observation. The shared conditioning network processes the projected fused features together with the diffusion-timestep token and positional embeddings to produce memory tokens $\mathbf{m}$. The decoder receives the noisy action sequence through shared action and positional embeddings. Dropout is omitted from the notation, and the base-policy attention masks are retained.

Attention-output aggregation. Given the fused observation representation, the decoder next integrates the pretrained attention modules. Although the attention modules are structurally compatible, directly replacing them with a single averaged module can discard expert-specific interaction patterns. Rather than averaging attention parameters, we retain frozen expert branches and learn to aggregate their outputs:

$$\mathbf{u} = \mathbf{x} + \sum_{i=1}^{N} \alpha_i \,\text{Self-Attn}_i(\mathbf{x}), \tag{2}$$

$$\mathbf{v} = \mathbf{u} + \sum_{i=1}^{N} \alpha_i \,\text{Cross-Attn}_i(\mathbf{u}, \mathbf{m}). \tag{3}$$

Here, $\mathbf{x}$ denotes the incoming action tokens, and Self-Attn_i and Cross-Attn_i denote expert i's self-attention and cross-attention operations, each preceded by its corresponding shared layer normalization. Cross-attention uses the updated tokens $\mathbf{u}$ as queries and memory $\mathbf{m}$ as keys and values. The weights α_i are softmax-normalized and shared between self- and cross-attention within a layer, but differ across layers. They are learned globally rather than conditioned on the input. Both observation-fusion and attention-mixing logits are initialized to zero, giving uniform weights at the start of consolidation.

Heterogeneous FFN routing. The FFNs require a different treatment. Unlike the attention modules, they may have different hidden widths and therefore cannot be combined through parameter averaging. We instead preserve each pretrained FFN at its original width and expose them as a heterogeneous expert set. After attention, we normalize $\mathbf{v}$ to obtain $\mathbf{h}$ and average its action tokens into one sequence representation. A trainable linear router followed by softmax scores the FFN experts from this representation. It selects the top-k experts and renormalizes their probabilities to sum to one. The layer output is

$$\mathbf{y} = \mathbf{v} + \sum_{i \in \mathcal{S}} g_i \, \mathrm{FFN}_i(\mathbf{h}), \tag{4}$$

where $\mathcal{S}$ contains the selected experts and g_i is the normalized routing weight, with $1 \leq k \leq N$. All tokens in an action sequence share the same selection and weights, which can vary across samples, layers, and denoising steps. The shared prediction head maps the final normalized decoder output to the predicted noise. Sparse routing reduces the number of FFN evaluations; all encoder and attention branches remain active.

3.3 Optimization and Evaluation

We train experts and the consolidated policy to predict Gaussian noise added to demonstration actions at uniformly sampled diffusion timesteps. The noise level follows the diffusion schedule. The denoising loss $\mathcal{L}_{\mathrm{diff}}$ is the mean squared error between the added and predicted noise, averaged over action steps, action dimensions, demonstrations, timesteps, and noise samples. Consolidation additionally uses a load-balancing term:

$$\mathcal{L} = \mathcal{L}_{\mathrm{diff}} + \eta \, \mathcal{L}_{\mathrm{bal}}. \tag{5}$$

For each layer, the balancing term is N times the dot product of two minibatch statistics: the fraction of sequences assigning the highest routing probability to each expert, and the mean routing probability of each expert before top-k selection. We average this term across layers and treat the assignment fractions as constants during differentiation. The assignment statistic uses only the highest-scoring expert, even when forward routing activates multiple experts. The coefficient $\eta \geq 0$ controls the regularization strength.

Experts minimize the denoising loss on their own demonstrations using their own encoder features. Consolidation minimizes the combined objective on pooled demonstrations using fused features. We optimize the fusion and attention weights, routers, FFNs, and shared components, while allowing gradients to pass through the frozen attention modules to their inputs. The detailed training and consolidation algorithm is provided in Appendix A.3.

At inference, the policy iteratively denoises action sequences conditioned on the observation history. Cross-environment evaluation covers seen, held-out, and mixed domains, although training uses only demonstrations from the training atomic domains. The other settings assess consolidation across the included tasks and robots, not transfer to unseen tasks or embodiments.

4 Experiments

We evaluate whether Policy-Soup can consolidate specialized manipulation policies across environments, tasks, and embodiments. The experiments first test retention of source-domain capabilities and generalization to environmental shifts, then examine consolidation across included tasks and embodiments and deployment on physical robots. Finally, controlled comparisons assess FFN capacity allocation, expert initialization, and the sensitivity of a consolidated policy to post-hoc attention replacement.

4.1 Experimental Setup

Simulation Benchmarks. We evaluate Policy-Soup on three simulation benchmarks, collectively covering cross-environment, cross-task, and cross-embodiment consolidation. Figure 2 illustrates the atomic domains used for cross-environment evaluation.

- **RoboTwin 2.0** Chen et al. (2025) supports our evaluation across all three consolidation settings. For cross-environment consolidation, we use four tasks: *open laptop*, *shake bottle*, *place container plate*, and *lift pot*. Each task has four source experts specializing in lighting (L), table texture (T),

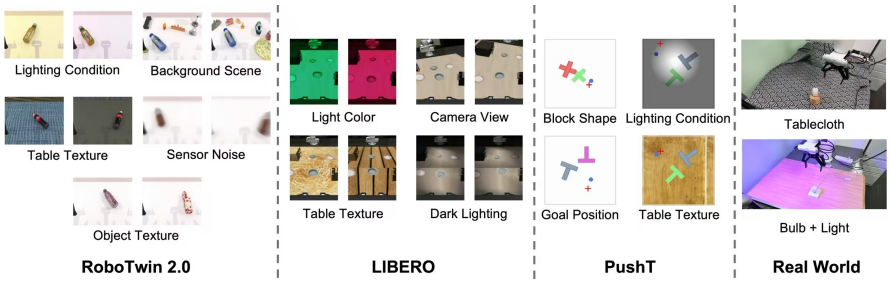

Figure 2: Atomic domains used in our experiments across three simulation benchmarks and real-world settings.

background (B), and sensor noise (S), respectively; object texture (O) is held out. Additional five-expert experiments on *grab roller* and *turn switch* include O as a training domain (Table 12). For cross-task consolidation, we combine three specialists for *place container plate*, *place empty cup*, and *place mouse pad*. For cross-embodiment consolidation, we combine three specialists trained on task *place container plate* using X5, ALOHA-AgileX, and Piper.

- **LIBERO** Liu et al. (2023) is used to evaluate cross-environment consolidation on the *open top drawer* task. The source domains vary light color (C), table texture (T), and camera viewpoint (V), with dark lighting (D) held out for evaluation.
- **PushT** Chi et al. (2024) is used to evaluate cross-environment consolidation under variations in block shape (B), lighting (L), and goal position (G). These factors define the source domains, while table texture (T) is held out for evaluation.

Real-World Robot Experiments. We evaluate POLICY-SOUP on physical robots in two separate three-expert consolidation experiments. For cross-environment consolidation, we use bulb tightening, with source experts specializing in tablecloth patterns, bulb appearance, and lighting condition. For cross-task consolidation, we design three manipulation tasks: *pick and place mouse*, *pick and place egg*, and *place the block onto stick*. The detailed experimental setup is provided in Section 4.5.

Training and evaluation. Experts have compatible observation, action, and layer interfaces but may differ in FFN hidden width. For cross-environment experiments, widths are fixed before expert training using preliminary performance-based difficulty assessments. POLICY-SOUP is trained on pooled source demonstrations following Section 3. RoboTwin 2.0, LIBERO, and real-world experiments report success rates as proportions. PushT instead reports final-state intersection-over-union (IoU), separately from success rate. Aggregate entries are unweighted means over the indicated conditions, tasks, or embodiments. More implementation details are provided in Appendix A.

4.2 CROSS-ENVIRONMENT CONSOLIDATION AND GENERALIZATION

Retaining source-domain performance. Table 1 summarizes the original domain-level results, with the full matrices retained in Appendix B. POLICY-SOUP exceeds or matches the best-expert oracle in every evaluated seen atomic domain across the three benchmarks. These results show that a single consolidated policy can retain and, in these settings, improve upon the capabilities of its source experts.

Generalization beyond the training domains. Beyond improvements on seen source domains, POLICY-SOUP achieves better performance on held-out domains and mixed configurations. For instance, on mixed configurations, mean success increases from 0.471 to 0.690 for RoboTwin 2.0 *shake bottle* and from 0.225 to 0.470 for LIBERO. Together, these results support generalization beyond the training domains to the tested held-out factors and factor combinations.

4.3 CROSS-TASK CONSOLIDATION

The preceding results demonstrate POLICY-SOUP's ability to generalize across environmental domains while keeping the task fixed. We next evaluate its performance in cross-task settings by consolidating specialists trained on different tasks. The source specialists are trained on RoboTwin 2.0 *place container plate*, *place empty cup*, and *place mouse pad*, respectively. As shown in Table 2(a),

Table 1: **Cross-environment consolidation and generalization.** RoboTwin 2.0 and LIBERO report success rates (100 trials per policy and configuration); PushT reports IoU, not success rate. Means are unweighted over configurations. RoboTwin 2.0 averages four seen domains and seven mixed configurations (six mixed-seen and one mixed+held-out); LIBERO and PushT each average three seen domains and four mixed-seen configurations. Held-out reports one atomic domain per benchmark. The best-expert oracle takes the maximum over all source experts per configuration before averaging, which is not a single deployable policy. Higher is better for both metrics. Bold numbers indicate the best performance.

Task	Policy	Seen	Held-out	Mixed
RoboTwin 2.0: ***success rate*** ↑				
Open laptop	Best-expert oracle	0.508	0.640	0.311
	POLICY-SOUP	**0.575**	**0.650**	**0.463**
Shake bottle	Best-expert oracle	0.808	0.920	0.471
	POLICY-SOUP	**0.850**	**0.960**	**0.690**
Place container plate	Best-expert oracle	0.280	0.320	0.050
	POLICY-SOUP	**0.498**	**0.610**	**0.179**
Lift pot	Best-expert oracle	0.568	0.140	0.126
	POLICY-SOUP	**0.580**	**0.180**	**0.139**
LIBERO: ***success rate*** ↑				
Open top drawer	Best-expert oracle	0.540	0.140	0.225
	POLICY-SOUP	**0.627**	**0.560**	**0.470**
PushT: ***IoU*** ↑				
PushT	Best-expert oracle	0.738	0.133	0.278
	POLICY-SOUP	**0.844**	**0.171**	**0.699**

Table 2: **Task and embodiment consolidation on RoboTwin 2.0.** Panels are separate three-expert experiments. Success rates (↑): 100 trials per policy. Expert subscripts identify the source task in panel (a) (plate, cup, pad) or robot in panel (b) (X5, AgileX, Piper), following the column order; AgileX denotes ALOHA-AgileX. Bold numbers indicate the best performance.

(a) Cross-task

Method	Place container plate	Place empty cup	Place mouse pad	Mean ↑
$\text{Expert}_{\text{plate}}$	0.38	0.00	0.00	0.127
$\text{Expert}_{\text{cup}}$	0.00	0.28	0.00	0.093
$\text{Expert}_{\text{pad}}$	0.00	0.00	0.01	0.003
POLICY-SOUP	**0.62**	**0.32**	**0.02**	**0.320**

(b) Cross-embodiment

Method	X5	ALOHA-AgileX	Piper	Mean ↑
$\text{Expert}_{\text{X5}}$	0.41	0.00	0.00	0.137
$\text{Expert}_{\text{AgileX}}$	0.00	0.68	0.00	0.227
$\text{Expert}_{\text{Piper}}$	0.00	0.00	0.56	0.187
POLICY-SOUP	**0.60**	**0.73**	**0.62**	**0.650**

mean success across the three tasks increases from 22.3% for the corresponding specialists to 32.0% for POLICY-SOUP. Success on *place container plate* increases from 38% to 62%, representing the largest improvement. Notably, each specialist achieves 0% success on the other two tasks, whereas POLICY-SOUP succeeds on all three using a single consolidated policy. These results show that one consolidated policy achieves higher observed success than the corresponding specialist on each of the three source tasks.

4.4 CROSS-EMBODIMENT CONSOLIDATION

The preceding experiments keep the robot embodiment fixed within each consolidation setting. We next evaluate whether POLICY-SOUP can also achieve strong performance across different robot embodiments. We combine three experts trained on task *place container plate* using X5, ALOHA-AgileX, and Piper. The same consolidated policy is evaluated on all three robots using compatible observation and action interfaces. Table 2(b) shows that POLICY-SOUP raises mean success from 55% to 65%. Success increases from 41% to 60% on X5, from 68% to 73% on ALOHA-AgileX, and from 56% to 62% on Piper. In the evaluation trials, each specialist achieves 0% success on the other two embodiments. In contrast, POLICY-SOUP successfully performs the task across all three embodiments using a single consolidated policy. These gains show that the consolidated policy

Table 3: **Real-world consolidation.** Success rates (↑) use 20 trials per policy and condition or task. Each panel uses a separate consolidated policy. In panel (a), $\text{Expert}_{\text{table}}$, $\text{Expert}_{\text{bulb}}$, and $\text{Expert}_{\text{light}}$ specialize in tablecloth patterns, bulb appearance, and lighting condition, respectively. Atomic factors are seen during training, while their tested combinations are mixed-seen conditions. In panel (b), $\text{Expert}_{\text{mouse}}$, $\text{Expert}_{\text{egg}}$, and $\text{Expert}_{\text{block}}$ specialize in *pick and place mouse*, *pick and place egg*, and *place block onto stick*, respectively. Bold numbers indicate the best performance.

(a) Cross-environment

Condition	Split	$\text{Expert}_{\text{table}}$	$\text{Expert}_{\text{bulb}}$	$\text{Expert}_{\text{light}}$	POLICY-SOUP
Tablecloth	Seen	0.40	0.10	0.00	**0.65**
Bulb Appearance	Seen	0.05	0.60	0.05	**0.80**
Lighting Condition	Seen	0.10	0.00	0.70	**0.75**
Table+Bulb	Mixed	0.05	0.10	0.00	**0.40**
Bulb+Lighting	Mixed	0.00	0.05	0.15	**0.60**
Table+Bulb+Lighting	Mixed	0.00	0.00	0.00	**0.15**

(b) Cross-task

Task	$\text{Expert}_{\text{mouse}}$	$\text{Expert}_{\text{egg}}$	$\text{Expert}_{\text{block}}$	POLICY-SOUP
Pick and place mouse	0.80	0.00	0.00	**0.90**
Pick and place egg	0.00	0.70	0.00	**0.75**
Place block onto stick	0.00	0.00	0.65	**0.70**

can retain and improve the task performance of its source specialists across the three evaluated embodiments.

4.5 REAL-WORLD EVALUATION

We evaluate POLICY-SOUP on physical robots in two separate three-expert experiments: bulb tightening under environmental variation and consolidation across three manipulation tasks (Table 3). We include video demonstrations of both experiments in the supplementary material.

Cross-environment consolidation. We design the task bulb tightening within three domains: tablecloth patterns, bulb appearance, and lighting condition. Evaluation covers individual factors and mixed-seen combinations excluded from consolidation training. POLICY-SOUP outperforms every source expert in all six tested conditions. With all three factors combined, POLICY-SOUP achieves 15% success, whereas no source expert succeeds in the reported trials.

Cross-task consolidation. We design three tasks: *pick and place mouse*, *pick and place egg*, and *place the block onto stick*. The consolidated policy achieves success rates of 90%, 75%, and 70%, compared with 80%, 70%, and 65% for the respective specialists. It performs all three tasks, whereas source experts record zero off-task success in the reported trials. Together, these results demonstrate that POLICY-SOUP effectively consolidates specialist capabilities on physical robots, improving performance across the evaluated environmental conditions or source tasks within each experiment.

4.6 ABLATION STUDIES

Heterogeneous FFN allocation at a fixed parameter budget. We examine whether assigning expert capacities according to domain difficulty and consolidating the resulting heterogeneous experts improves performance. To this end, we compare homogeneous and heterogeneous FFN allocations in a four-expert experiment on RoboTwin 2.0 *open laptop* (Table 4). In L/T/B/S order, the homogeneous and heterogeneous expert widths are $(1344, 1344, 1344, 1344)$ and $(1024, 1536, 1536, 1280)$, respectively. Both models contain 22.071 million FFN parameters and use the same training data, training budget, and top-2 routing. Heterogeneous allocation increases success on seen domains, held-out domain, and mixed conditions. These results support allocating more FFN parameters to experts for harder domains to improve POLICY-SOUP under a fixed total FFN parameter budget. These results also highlight the value of POLICY-SOUP's ability to consolidate heterogeneous experts.

Table 4: **Capacity-controlled FFN ablation on RoboTwin 2.0 *open laptop*.** Success rates (↑) use 100 trials per policy/configuration. Both allocations use 22.071M FFN parameters, identical training data and budgets, and top-2 routing. Bold numbers indicate the best performance.

Allocation	Seen	Held-out	Mixed
Homogeneous	0.545	0.640	0.390
Heterogeneous	**0.575**	**0.650**	**0.463**
Gain (percentage points)	+3.0	+1.0	+7.3

Table 5: **Expert initialization.** Split means from Table 10 (↑). With/Without denotes the presence/absence of pretrained expert initialization under the same architecture, pooled data, and hyperparameters. Bold numbers indicate the best performance.

Initialization	Seen	Held-out	Mixed
RoboTwin 2.0: *success rate*			
Without	0.615	0.780	0.334
With	**0.674**	**0.785**	**0.434**
PushT: *IoU*			
Without	0.826	0.164	0.617
With	**0.844**	**0.171**	**0.699**

Expert initialization. We compare POLICY-SOUP with a variant trained without pretrained expert initialization, using the same architecture, pooled data, and hyperparameters. As shown in Table 5, expert initialization improves all three split means on both benchmarks, with the largest absolute gains on Mixed. On RoboTwin 2.0, success rates increase from 0.615 to 0.674 on Seen, from 0.780 to 0.785 on Held-out, and from 0.334 to 0.434 on Mixed. These results support the usefulness of pretrained expert parameters within the stated consolidation budget.

Comparison of model merging strategies. We compare POLICY-SOUP with four training-free model parameter merging methods, uniform WA Wortsman et al. (2022a), TA Ilharco et al. (2023), TIES Yadav et al. (2023a), and TSV Gargiulo et al. (2025) on two RoboTwin 2.0 tasks. We reimplement TA, TIES, and TSV in our experimental setting, applying them to the attention parameters of the consolidated checkpoints. Their setting-specific configurations and implementation details are provided in Appendix B.3.

Full results for *shake bottle* and *open laptop* are provided in Table 11. POLICY-SOUP achieves a mean success rate of 71.3%, compared with 27.1% for WA, the strongest alternative, a gain of 44.1 percentage points. It also achieves the highest task-averaged success rate under each of the four conditions. These results show that POLICY-SOUP outperforms the compared model parameter merging methods under the evaluated settings.

Table 6: **Model merging on RoboTwin 2.0.** Success rates (↑) average *shake bottle* and *open laptop* equally. Bold numbers indicate the best performance.

Method	L	T	B	S	Mean
POLICY-SOUP	**0.740**	**0.695**	**0.580**	**0.835**	**0.713**
WA	0.305	0.260	0.290	0.230	0.271
TA	0.180	0.230	0.265	0.210	0.221
TIES	0.165	0.225	0.225	0.210	0.206
TSV	0.160	0.145	0.190	0.140	0.159

5 CONCLUSION

We presented POLICY-SOUP, a framework for consolidating heterogeneous diffusion-policy experts into a unified visuomotor policy. It shares transferable knowledge while preserving expert specialization. Learned visual-feature fusion and layer-wise attention-output aggregation enable knowledge sharing, while sequence-level sparse routing retains heterogeneous FFNs at their original widths. Experiments on RoboTwin 2.0, LIBERO, PushT, and real-world manipulation show strong source-domain performance and improved robustness under tested environmental shifts. Separate cross-task and cross-embodiment evaluations further demonstrate consolidation of capabilities across the included tasks and robots. These findings support heterogeneous expert reuse as a route toward more generalizable manipulation. Nevertheless, compatible interfaces remain necessary, difficult tasks and combined shifts remain challenging, and scaling beyond the moderate-size diffusion policies studied here remains an open direction.

AI USE STATEMENT

In this work, we used large language models to assist with drafting and editing manuscript text, refining table captions, adjusting LaTeX formatting, and creating auxiliary scripts for document compilation and consistency checks. These uses were limited to manuscript preparation and did not involve developing the research methodology, formulating hypotheses or mathematical claims, designing experiments, implementing the proposed method, generating or processing research data, translating text, or interpreting experimental results. We reviewed the AI-assisted text against the underlying methods and results and checked the auxiliary scripts and their outputs for correctness. We take full responsibility for the accuracy, originality, and integrity of the final submission, including all text and artifacts prepared with LLM assistance.

REFERENCES

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Mengyuan Yan, and Andy Zeng. Do as i can, not as i say: Grounding language in robotic affordances, 2022. URL https://arxiv.org/abs/2204.01691.

Konstantinos Bousmalis, Giulia Vezzani, Dushyant Rao, Coline Devin, Alex X. Lee, Maria Bauza, Todor Davchev, Yuxiang Zhou, Agrim Gupta, Akhil Raju, Antoine Laurens, Claudio Fantacci, Valentin Dalibard, Martina Zambelli, Murilo Martins, Rugile Pevceviciute, Michiel Blokzijl, Misha Denil, Nathan Batchelor, Thomas Lampe, Emilio Parisotto, Konrad Żołna, Scott Reed, Sergio Gómez Colmenarejo, Jon Scholz, Abbas Abdolmaleki, Oliver Groth, Jean-Baptiste Regli, Oleg Sushkov, Tom Rothörl, José Enrique Chen, Yusuf Aytar, Dave Barker, Joy Ortiz, Martin Riedmiller, Jost Tobias Springenberg, Raia Hadsell, Francesco Nori, and Nicolas Heess. Robocat: A self-improving generalist agent for robotic manipulation, 2023. URL https://arxiv.org/abs/2306.11706.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-1: Robotics transformer for real-world control at scale. In *arXiv preprint arXiv:2212.06817*, 2022.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alex Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *arXiv preprint arXiv:2307.15818*, 2023.

Onur Celik, Dongzhuoran Zhou, Ge Li, Philipp Becker, and Gerhard Neumann. Specializing versatile skill libraries using local mixture of experts, 2022. URL https://arxiv.org/abs/2112.04216.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, Weiliang Deng, Yubin Guo, Tian Nian, Xuanbing Xie, Qiangyu Chen, Kailun Su, Tianling Xu, Guodong Liu, Mengkang Hu, Huan ang Gao, Kaixuan Wang, Zhixuan Liang, Yusen Qin, Xiaokang Yang, Ping Luo, and Yao Mu. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation, 2025. URL https://arxiv.org/abs/2506.18088.

Runxi Cheng, Feng Xiong, Yongxian Wei, Wanyun Zhu, and Chun Yuan. Whoever started the interference should end it: Guiding data-free model merging via task vectors, 2025. URL https://arxiv.org/abs/2503.08099.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion, 2024. URL https://arxiv.org/abs/2303.04137.

M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. Palm-e: An embodied multimodal language model, 2023. URL https://arxiv.org/abs/2303.03378.

Nan Du, Yanping Huang, Andrew M. Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, Barret Zoph, Liam Fedus, Maarten Bosma, Zongwei Zhou, Tao Wang, Yu Emma Wang, Kellie Webster, Marie Pellat, Kevin Robinson, Kathleen Meier-Hellstern, Toju Duke, Lucas Dixon, Kun Zhang, Quoc V Le, Yonghui Wu, Zhifeng Chen, and Claire Cui. Glam: Efficient scaling of language models with mixture-of-experts, 2022. URL https://arxiv.org/abs/2112.06905.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity, 2022. URL https://arxiv.org/abs/2101.03961.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, Jinlan Fu, Jingjing Gong, and Xipeng Qiu. Libero-plus: In-depth robustness analysis of vision-language-action models, 2025. URL https://arxiv.org/abs/2510.13626.

Yuxia Fu, Zhizhen Zhang, Yuqi Zhang, Zijian Wang, Zi Huang, and Yadan Luo. Mergevla: Cross-skill model merging toward a generalist vision-language-action agent, 2026. URL https://arxiv.org/abs/2511.18810.

Zipeng Fu, Tony Z. Zhao, and Chelsea Finn. Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation, 2024. URL https://arxiv.org/abs/2401.02117.

Jensen Gao, Suneel Belkhale, Sudeep Dasari, Ashwin Balakrishna, Dhruv Shah, and Dorsa Sadigh. A taxonomy for evaluating generalist robot manipulation policies, 2026. URL https://arxiv.org/abs/2503.01238.

Antonio Andrea Gargiulo, Donato Crisostomi, Maria Sofia Bucarelli, Simone Scardapane, Fabrizio Silvestri, and Emanuele Rodolà. Task singular vectors: Reducing task interference in model merging. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18695–18705, 2025. doi: 10.1109/CVPR52734.2025.01742.

Ce Hao, Xuanran Zhai, Yaohua Liu, and Harold Soh. Abstracting robot manipulation skills via mixture-of-experts diffusion policies, 2026. URL https://arxiv.org/abs/2601.21251.

Chenyu Huang, Peng Ye, Tao Chen, Tong He, Xiangyu Yue, and Wanli Ouyang. Emr-merging: Tuning-free high-performance model merging, 2024. URL https://arxiv.org/abs/2405.17461.

Suning Huang, Zheyu Zhang, Tianhai Liang, Yihan Xu, Zhehao Kou, Chenhao Lu, Guowei Xu, Zhengrong Xue, and Huazhe Xu. Mentor: Mixture-of-experts network with task-oriented perturbation for visual reinforcement learning, 2025. URL https://arxiv.org/abs/2410.14972.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic, 2023. URL https://arxiv.org/abs/2212.04089.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization, 2019. URL https://arxiv.org/abs/1803.05407.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991. doi: 10.1162/neco.1991.3.1.79.

Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning, 2022. URL https://arxiv.org/abs/2202.02005.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mixtral of experts, 2024. URL https://arxiv.org/abs/2401.04088.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag 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, 2024. URL https://arxiv.org/abs/2406.09246.

Dmytro Kuzmenko and Nadiya Shvai. Moira: Modular instruction routing architecture for multi-task robotics. *Neurocomputing*, 674:132962, April 2026. ISSN 0925-2312. doi: 10.1016/j.neucom.2026.132962. URL http://dx.doi.org/10.1016/j.neucom.2026.132962.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning, 2023. URL https://arxiv.org/abs/2306.03310.

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, 2017. URL https://arxiv.org/abs/1701.06538.

Sainbayar Sukhbaatar, Olga Golovneva, Vasu Sharma, Hu Xu, Xi Victoria Lin, Baptiste Rozière, Jacob Kahn, Daniel Li, Wen tau Yih, Jason Weston, and Xian Li. Branch-train-mix: Mixing expert llms into a mixture-of-experts llm, 2024. URL https://arxiv.org/abs/2403.07816.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy, 2024. URL https://arxiv.org/abs/2405.12213.

Mitchell Wortsman, Gabriel Ilharco, Samir Yitzhak Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S. Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time, 2022a. URL https://arxiv.org/abs/2203.05482.

Mitchell Wortsman, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo-Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, and Ludwig Schmidt. Robust fine-tuning of zero-shot models, 2022b. URL https://arxiv.org/abs/2109.01903.

Tian-Yu Xiang, Ao-Qun Jin, Xiao-Hu Zhou, Mei-Jiang Gui, Xiao-Liang Xie, Shi-Qi Liu, Shuang-Yi Wang, Sheng-Bin Duan, Fu-Chao Xie, Wen-Kai Wang, Si-Cheng Wang, Ling-Yun Li, Tian Tu, and Zeng-Guang Hou. Parallels between vla model post-training and human motor learning: Progress, challenges, and trends, 2026. URL https://arxiv.org/abs/2506.20966.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. TIES-merging: Resolving interference when merging models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a. URL https://openreview.net/forum?id=xtaX3WyCj1.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. Ties-merging: Resolving interference when merging models, 2023b. URL https://arxiv.org/abs/2306.01708.

Prateek Yadav, Colin Raffel, Mohammed Muqeeth, Lucas Caccia, Haokun Liu, Tianlong Chen, Mohit Bansal, Leshem Choshen, and Alessandro Sordoni. A survey on model moerging: Recycling and routing among specialized experts for collaborative learning, 2025. URL https://arxiv.org/abs/2408.07057.

Yajat Yadav, Zhiyuan Zhou, Andrew Wagenmaker, Karl Pertsch, and Sergey Levine. Robust fine-tuning of vision-language-action robot policies via parameter merging, 2026. URL https://arxiv.org/abs/2512.08333.

Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li. Language models are super mario: Absorbing abilities from homologous models as a free lunch, 2024. URL https://arxiv.org/abs/2311.03099.

Ruiqi Zhu, Endong Sun, Guanhe Huang, and Oya Celiktutan. Efficient continual adaptation of pretrained robotic policy with online meta-learned adapters, 2025. URL https://arxiv.org/abs/2503.18684.

A Implementation Details

A.1 Policy Architecture and Training

Base Policy Architecture. All experts and POLICY-SOUP use a Diffusion Transformer image policy Chi et al. (2024). Each camera has an independent ResNet-18 encoder with Group Normalization, ImageNet normalization, and random cropping during training. The decoder has eight layers, four attention heads, and embedding dimension $d = 256$. Experts share decoder depth, token dimension, attention configuration, and observation-feature shape. Their FFNs have compatible input/output dimensions but may differ in hidden width. Widths are fixed before expert training, guided by preliminary performance-based difficulty assessments in the cross-environment setting.

We use DDPM epsilon prediction with 100 training timesteps, a `squaredcos_cap_v2` schedule, $\beta_{\min} = 0.0001$, and $\beta_{\max} = 0.02$. Inference uses 100 denoising steps. The policy predicts eight action steps from three observation steps and executes six action steps per call.

Expert Policy Training. For each task and atomic environment, we jointly train an expert's encoder and policy, starting from a shared observation encoder pre-trained with augmented data. Training uses AdamW with learning rate 3×10^{-5}, $\beta = (0.95, 0.999)$, weight decay 10^{-6}, cosine annealing with 500 warm-up steps, batch size 128, and 1,000 epochs. For each RoboTwin 2.0 task and each source atomic domain, we train the corresponding domain-specialized expert using 100 demonstration trajectories. The demonstrations are treated as complete episodes, and frames from the same trajectory are kept in the same data split. For LIBERO, we use the original 50 demonstrations provided by LIBERO-90 for each task and do not collect additional demonstrations. Instead, we replay selected original demonstrations under different environment configurations and save the resulting domain-specific visual observations. The original action sequences and task labels are retained, whereas the visual observations are regenerated under the target environment configuration.

The FFN expansion range of $3.0\times$–$6.0\times$ applies only to RoboTwin 2.0. With $d = 256$, this gives task-specific hidden dimensions of 768–1,536. LIBERO and PushT use the benchmark-specific widths listed in Tables 8 and 9, respectively.

POLICY-SOUP Training. POLICY-SOUP consolidates $N = 3$–5 pre-trained experts, depending on the task. It learns to aggregate frozen attention-branch outputs and retains heterogeneous FFNs. For *open laptop*, the light, table, background, and sensor-noise FFN hidden dimensions are $[1024, 1536, 1536, 1280]$, respectively. Each decoder layer uses top-k routing with $k = 2$ and a Switch Transformer-style load-balancing loss with coefficient $\eta = 0.01$.

Visual fusion uses a *Weighted Ensemble ObsEncoder* with N frozen encoders and uniformly initialized learnable mixing logits $\boldsymbol{\lambda} \in \mathbb{R}^N$:

$$\mathbf{z} = \sum_{i=1}^{N} \mathrm{softmax}(\boldsymbol{\lambda})_i \, \mathrm{Enc}_i(\mathbf{o}).$$

Expert observation encoders and self- and cross-attention parameters remain frozen. Trainable parameters comprise the FFNs, routers, attention-aggregation and observation-fusion logits, layer normalizations, shared input/output embeddings, and action prediction head.

Consolidation uses the pooled atomic-domain data and eight-GPU PyTorch Distributed Data Parallel via `torchrun`. We use AdamW with learning rate 10^{-4}, $\beta = (0.95, 0.999)$, weight decay 10^{-6}, cosine annealing with 500 warm-up steps, batch size 128, and 2,500 epochs. The parameter exponential moving average has maximum decay $\gamma_{\max} = 0.9999$.

A.2 RoboTwin 2.0 Domain Randomization

The five atomic perturbations are configured independently in RoboTwin's SAPIEN-based simulator. The common *clean* reference disables randomization, fixes table height, and uses a uniform white background. The mixed settings *L_T_B_S* and *L_T_B_S_O* combine lighting, table texture, background, and sensor noise; the latter also includes object texture.

Lighting Condition. At the beginning of each episode, the RGB intensities of all directional and point lights in the scene are independently resampled from a uniform distribution over $[0, 1]^3$. The

scene contains one directional light, whose default direction is $[0, 0.5, -1]$, and two point lights located at $[\pm 1, 0, 1.8]$. This randomization induces substantial photometric variation, including monochromatic color shifts, strong illumination changes, and near-dark visual conditions. In addition, with probability 0.02 per episode, we activate an extreme illumination mode, in which all point lights undergo large simultaneous photometric shifts.

Background Scene. At the beginning of each episode, up to ten distractor objects are sampled from a curated pool of 3D mesh assets covering diverse object categories, such as household items and tools, and placed on the tabletop. Each distractor is assigned a random valid position within the table boundary, $x \in [-0.59, 0.59]\,\mathrm{m}$ and $y \in [-0.34, 0.34]\,\mathrm{m}$, together with a random in-plane rotation. We explicitly avoid overlap between distractors and task-relevant objects using radius-based collision checking. To introduce mild geometric variation in the workspace, the table height is additionally perturbed by an episode-wise random offset sampled uniformly from $[-0.03, 0]\,\mathrm{m}$.

Table Texture. At the beginning of each episode, the visual textures of the wall and tabletop surface are replaced with images sampled from a held-out texture library located at `assets/background_texture/`. During training-data collection, textures are drawn from the *seen* split, whereas evaluation on unseen domains uses the *unseen* split. The wall and tabletop textures are sampled independently, yielding a large combinatorial space of workspace appearances. With probability 0.02 per episode, the clean, untextured appearance is retained for each surface, preventing the policy from observing only fully textured scenes. The same table-height perturbation used in the *table* domain, i.e., a uniform offset in $[-0.03, 0]\,\mathrm{m}$, is also applied in this domain.

Object Texture. Object-level appearance variation is introduced by replacing the base-color texture of the primary task-relevant object, such as the laptop in *open_laptop* or the bottle in *shake_bottle*, with a randomly sampled image from the Describable Textures Dataset (DTD) Cimpoi et al. (2014). We use ten DTD semantic categories, including examples such as *banded*, *dotted*, and *woven*. Images from all selected categories are pooled, and one image is sampled uniformly at random for each episode. Texture replacement is implemented by setting the base-color texture of all render shapes associated with the target actor to the selected DTD image via SAPIEN's `RenderTexture2D` interface. The material tint is reset to neutral white so that the sampled texture is rendered without additional color bias.

Sensor Noise. We apply post-rendering photometric corruptions to the RGB observations from all cameras, following the LIBERO-plus sensor-noise protocol Fei et al. (2025). The protocol defines five noise types, denoted N1–N5, each with five intensity levels. We use level 3, corresponding to moderate corruption, throughout all experiments. At the beginning of each episode, one of the following noise types is sampled uniformly at random and applied directly to the rendered `uint8` RGB images before they are provided to the policy.

- **N1 Motion Blur.** A one-dimensional Gaussian kernel with radius $r = 15$ and standard deviation $\sigma = 10$ is placed along the center row of a 15×15 matrix. The kernel is rotated by an angle $\theta \sim \mathcal{U}(0^\circ, 180^\circ)$ and convolved independently with each RGB channel.
- **N2 Gaussian Blur.** Isotropic Gaussian blur is applied to the image with standard deviation $\sigma = 5$ pixels along both spatial axes.
- **N3 Zoom Blur.** The corrupted image is obtained by averaging a sequence of rescaled images with scale factors $s \in [1.0, 1.34]$ using a step size of 0.02. Each rescaled image is center-cropped back to the original resolution, producing a radial blur effect.
- **N4 Fog.** Atmospheric haze is simulated using a Beer–Lambert transmission model:

$$I'(x, y) = I(x, y) \exp(-\alpha d^\beta) + \mathbf{c}_{\text{fog}} \left(1 - \exp(-\alpha d^\beta)\right),$$

  where d denotes a normalized radial depth proxy, $\alpha = 2.5$, $\beta = 2.0$, and $\mathbf{c}_{\text{fog}} = [210, 210, 210]$.
- **N5 Glass Blur.** Glass-like distortion is generated by first applying Gaussian blur with $\sigma = 1.5$, followed by two iterations of pixel-coordinate remapping with random spatial offsets $\Delta x, \Delta y \sim \mathcal{U}(-3, 3)$ pixels.

A.3 CONSOLIDATION ALGORITHM

Algorithm 1 follows Section 3: Stage 1 trains source experts on their respective demonstration sets $\mathcal{B}_i$, and Stage 2 consolidates them on pooled demonstrations. The source domains may represent environments, tasks, or embodiments. Let L be the decoder depth and K the number of diffusion timesteps; c_τ and σ_τ are the signal and noise coefficients of the prescribed diffusion schedule. Forward operations are applied independently to each sequence in a minibatch of size M; the sequence index b is shown only in the balancing statistics. $\mathrm{ActionEmbed}$ includes action and positional embeddings, and Cond constructs memory from projected fused features, the timestep token, and positional embeddings. Attention operators include their preceding shared layer normalizations, as in Section 3, and retain the base-policy masks. MSE averages over all batch, action-step, and action-dimension entries.

Algorithm 1 POLICY-SOUP: expert training and heterogeneous consolidation

Require: Demonstration sets $\{\mathcal{B}_i\}_{i=1}^N$; fixed FFN widths $\{d_{\mathrm{f}}^{(i)}\}_{i=1}^N$; top-$k$ with $1 \le k \le N$; balancing coefficient η; training budgets $S_{\mathrm{exp}}, S_{\mathrm{con}}$
Ensure: Unified policy π_{uni}

Stage 1: Source-expert training
for $i = 1, \dots, N$ **do**
  Initialize expert π_i with encoder Enc_i and FFN width $d_{\mathrm{f}}^{(i)}$
  for $s = 1, \dots, S_{\mathrm{exp}}$ **do**
    Sample a minibatch $(\mathbf{o}, \mathbf{a})$ from $\mathcal{B}_i$
    Sample $\tau \sim \mathrm{Uniform}\{1, \dots, K\}$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ per sequence
    $\tilde{\mathbf{a}}_\tau \leftarrow c_\tau \mathbf{a} + \sigma_\tau \boldsymbol{\epsilon}$; $\mathbf{z}_i \leftarrow \mathrm{Enc}_i(\mathbf{o})$
    $\hat{\boldsymbol{\epsilon}} \leftarrow$ expert i's noise prediction given $(\tilde{\mathbf{a}}_\tau, \tau, \mathbf{z}_i)$
    Update all expert parameters, including Enc_i, to minimize $\mathrm{MSE}(\hat{\boldsymbol{\epsilon}}, \boldsymbol{\epsilon})$
  end for
end for
Stage 2: Heterogeneous policy consolidation
$\mathcal{B} \leftarrow \bigcup_{i=1}^N \mathcal{B}_i$; $\Theta_{\mathrm{sh}} \leftarrow \frac{1}{N} \sum_{i=1}^N \Theta_{\mathrm{sh},i}$ for compatible shared components
Retain all expert encoders, attention branches, and FFNs; freeze encoders and attention
Keep FFNs trainable at their original widths; initialize routers $\{\mathbf{R}^{(\ell)}\}_{\ell=1}^L$
Initialize fusion logits $\boldsymbol{\lambda} \leftarrow \mathbf{0}$ and attention logits $\boldsymbol{\rho}^{(\ell)} \leftarrow \mathbf{0}$ for all ℓ
for $s = 1, \dots, S_{\mathrm{con}}$ **do**
  Sample a minibatch $(\mathbf{o}, \mathbf{a})$ from $\mathcal{B}$
  Sample $\tau \sim \mathrm{Uniform}\{1, \dots, K\}$ and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ per sequence
  $\tilde{\mathbf{a}}_\tau \leftarrow c_\tau \mathbf{a} + \sigma_\tau \boldsymbol{\epsilon}$; $\mathbf{w} \leftarrow \mathrm{softmax}(\boldsymbol{\lambda})$; $\mathbf{z} \leftarrow \sum_{i=1}^N w_i \mathrm{Enc}_i(\mathbf{o})$
  $\mathbf{m} \leftarrow \mathrm{Cond}_{\Theta_{\mathrm{sh}}}(\mathbf{z}, \tau)$; $\mathbf{x}^{(1)} \leftarrow \mathrm{ActionEmbed}_{\Theta_{\mathrm{sh}}}(\tilde{\mathbf{a}}_\tau)$
  for $\ell = 1, \dots, L$ **do**
    $\boldsymbol{\alpha}^{(\ell)} \leftarrow \mathrm{softmax}(\boldsymbol{\rho}^{(\ell)})$
    $\mathbf{u}^{(\ell)} \leftarrow \mathbf{x}^{(\ell)} + \sum_{i=1}^N \alpha_i^{(\ell)}$ Self-Attn$_i^{(\ell)}(\mathbf{x}^{(\ell)})$
    $\mathbf{v}^{(\ell)} \leftarrow \mathbf{u}^{(\ell)} + \sum_{i=1}^N \alpha_i^{(\ell)}$ Cross-Attn$_i^{(\ell)}(\mathbf{u}^{(\ell)}, \mathbf{m})$
    $\mathbf{h}^{(\ell)} \leftarrow \mathrm{LN}_{\mathrm{ffn}}^{(\ell)}(\mathbf{v}^{(\ell)})$; $\bar{\mathbf{h}}^{(\ell)} \leftarrow \mathrm{Mean}_{\text{action tokens}}(\mathbf{h}^{(\ell)})$
    $\mathbf{p}^{(\ell)} \leftarrow \mathrm{softmax}(\mathbf{R}^{(\ell)} \bar{\mathbf{h}}^{(\ell)})$; $\mathcal{S}^{(\ell)} \leftarrow \text{Top-}k(\mathbf{p}^{(\ell)})$
    $g_i^{(\ell)} \leftarrow p_i^{(\ell)} / \sum_{j \in \mathcal{S}^{(\ell)}} p_j^{(\ell)}$ for $i \in \mathcal{S}^{(\ell)}$
    $\mathbf{x}^{(\ell+1)} \leftarrow \mathbf{v}^{(\ell)} + \sum_{i \in \mathcal{S}^{(\ell)}} g_i^{(\ell)} \mathrm{FFN}_i^{(\ell)}(\mathbf{h}^{(\ell)})$
    $f_i^{(\ell)} \leftarrow \frac{1}{M} \sum_{b=1}^M \mathbb{I}[i = \arg\max_j p_{b,j}^{(\ell)}]$; $\bar{p}_i^{(\ell)} \leftarrow \frac{1}{M} \sum_{b=1}^M p_{b,i}^{(\ell)}$ for all i
  end for
  $\hat{\boldsymbol{\epsilon}} \leftarrow \mathrm{Head}_{\Theta_{\mathrm{sh}}}(\mathrm{LN}_{\mathrm{out}}(\mathbf{x}^{(L+1)}))$; $\mathcal{L}_{\mathrm{diff}} \leftarrow \mathrm{MSE}(\hat{\boldsymbol{\epsilon}}, \boldsymbol{\epsilon})$
  $\mathcal{L}_{\mathrm{bal}} \leftarrow \frac{N}{L} \sum_{\ell=1}^L \sum_{i=1}^N \mathrm{stopgrad}(f_i^{(\ell)})\, \bar{p}_i^{(\ell)}$; $\mathcal{L} \leftarrow \mathcal{L}_{\mathrm{diff}} + \eta \mathcal{L}_{\mathrm{bal}}$
  Update fusion/attention logits, routers, FFNs, and Θ_{sh} to minimize $\mathcal{L}$
  Keep encoder/attention parameters frozen; backpropagate through attention to its inputs
end for
return π_{uni}; at inference, iteratively denoise actions conditioned on the observation history

B SUPPLEMENTARY EXPERIMENTAL RESULTS

B.1 CROSS-ENVIRONMENT RESULTS

Tables 7–9 provide the domain-level results underlying the main-text summaries. Each source expert is a separate model; all entries in a POLICY-SOUP row refer to its task-specific consolidated model. Domain symbols are local to each benchmark.

Table 7: **Full RoboTwin 2.0 cross-environment results.** Success rates use 100 rollouts per task–configuration. L, T, B, S, and O denote lighting, table texture, background, sensor noise, and object texture. L/T/B/S are seen; O is held out. The first six mixed columns combine seen factors, while L+T+B+S+O additionally contains the held-out factor. Underlining identifies best individual-expert scores.

Open Laptop	**Seen Domains**				**Held-out**	**Mixed Domains**						
	L	**T**	**B**	**S**	**O**	**L+T**	**L+B**	**B+S**	**L+T +B**	**L+B +S**	**L+T +B+S**	**L+T+B +S+O**
Exp 1 on L (d_f=4d)	<u>0.52</u>	0.04	0.08	0.41	0.54	0.01	0.03	0.06	0.00	0.02	0.00	0.00
Exp 2 on T (d_f=6d)	0.20	<u>0.41</u>	0.12	0.23	0.26	<u>0.37</u>	0.08	0.08	0.17	0.03	<u>0.38</u>	0.33
Exp 3 on B (d_f=6d)	0.30	0.40	<u>0.42</u>	0.15	0.49	0.29	<u>0.34</u>	<u>0.10</u>	<u>0.21</u>	<u>0.37</u>	0.33	<u>0.41</u>
Exp 4 on S (d_f=5d)	0.17	0.06	0.06	<u>0.68</u>	<u>0.64</u>	0.02	0.01	0.02	0.04	0.00	0.00	0.00
Policy-Soup	**0.56**	**0.50**	**0.54**	**0.70**	**0.65**	**0.38**	**0.70**	**0.37**	**0.41**	**0.50**	**0.43**	**0.45**

Shake Bottle	**Seen Domains**				**Held-out**	**Mixed Domains**						
	L	**T**	**B**	**S**	**O**	**L+T**	**L+B**	**B+S**	**L+T +B**	**L+B +S**	**L+T +B+S**	**L+T+B +S+O**
Exp 1 on L (d_f=3d)	<u>0.89</u>	0.36	0.37	0.76	0.83	0.41	0.38	0.16	0.22	0.30	0.16	0.00
Exp 2 on T (d_f=5d)	0.61	<u>0.87</u>	0.33	0.37	0.41	<u>0.84</u>	0.37	0.25	<u>0.36</u>	0.31	<u>0.33</u>	<u>0.39</u>
Exp 3 on B (d_f=5d)	0.59	0.28	<u>0.54</u>	0.55	0.75	0.24	0.39	<u>0.47</u>	0.26	<u>0.47</u>	0.23	0.06
Exp 4 on S (d_f=4d)	0.40	0.30	0.22	<u>0.93</u>	<u>0.92</u>	0.32	<u>0.44</u>	0.25	0.18	0.18	0.25	0.00
Policy-Soup	**0.92**	**0.89**	**0.62**	**0.97**	**0.96**	**0.90**	**0.64**	**0.90**	**0.62**	**0.79**	**0.50**	**0.48**

Place container plate	**Seen Domains**				**Held-out**	**Mixed Domains**						
	L	**T**	**B**	**S**	**O**	**L+T**	**L+B**	**B+S**	**L+T +B**	**L+B +S**	**L+T +B+S**	**L+T+B +S+O**
Exp 1 on L (d_f=4d)	<u>0.26</u>	0.00	0.00	0.29	0.21	0.00	0.00	0.00	0.01	0.00	0.00	0.00
Exp 2 on T (d_f=6d)	0.02	<u>0.24</u>	0.00	0.00	0.00	<u>0.20</u>	0.02	0.03	<u>0.04</u>	0.00	<u>0.01</u>	0.00
Exp 3 on B (d_f=6d)	0.01	0.02	<u>0.30</u>	0.24	0.23	0.02	<u>0.05</u>	<u>0.04</u>	0.00	<u>0.01</u>	0.00	0.00
Exp 4 on S (d_f=5d)	0.02	0.01	0.00	<u>0.32</u>	<u>0.32</u>	0.00	0.00	0.02	0.00	0.00	0.00	0.00
Policy-Soup	**0.56**	**0.31**	**0.52**	**0.60**	**0.61**	**0.26**	**0.47**	**0.20**	**0.21**	**0.05**	**0.04**	**0.02**

Lift Pot	**Seen Domains**				**Held-out**	**Mixed Domains**						
	L	**T**	**B**	**S**	**O**	**L+T**	**L+B**	**B+S**	**L+T +B**	**L+B +S**	**L+T +B+S**	**L+T+B +S+O**
Exp 1 on L (d_f=4d)	<u>0.59</u>	0.02	0.00	0.01	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Exp 2 on T (d_f=6d)	0.19	<u>0.34</u>	0.05	0.02	0.02	<u>0.40</u>	0.02	0.01	<u>0.25</u>	0.00	0.00	0.00
Exp 3 on B (d_f=6d)	0.44	0.00	<u>0.40</u>	0.02	<u>0.14</u>	0.03	<u>0.20</u>	<u>0.03</u>	0.00	0.00	0.00	0.00
Exp 4 on S (d_f=5d)	0.10	0.00	0.04	<u>0.94</u>	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Policy-Soup	**0.61**	**0.35**	**0.42**	**0.94**	**0.18**	**0.41**	**0.21**	**0.06**	**0.25**	**0.02**	**0.02**	**0.00**

Table 8: **Full LIBERO open-top-drawer results.** Success rates use 100 rollouts per configuration. C, T, V, and D denote light color, table texture, camera viewpoint, and dark lighting. C/T/V are seen, D is held out, and all mixed columns combine seen factors.

Open Drawer	**Seen Domains**			**Held-out**	**Mixed Domains**			
	C	**T**	**V**	**D**	**C+T**	**T+V**	**C+V**	**C+T+V**
Exp 1 on C (d_f=$2d$)	0.82	0.42	0.12	0.14	0.50	0.02	0.14	0.12
Exp 2 on T (d_f=$6d$)	0.10	0.52	0.16	0.06	0.00	0.12	0.08	0.06
Exp 3 on V (d_f=$10d$)	0.06	0.28	0.28	0.00	0.00	0.14	0.00	0.00
Policy-Soup	**0.84**	**0.64**	**0.40**	**0.56**	**0.72**	**0.24**	**0.54**	**0.38**

Table 9: **Full PushT results (IoU).** B, L, G, and T denote block shape, lighting, goal position, and table texture. B/L/G are seen, T is held out, and mixed columns combine seen factors. Higher IoU is better.

PushT	**Seen Domains**			**Held-out**	**Mixed Domains**			
	B	**L**	**G**	**T**	**B+L**	**L+G**	**B+G**	**B+L+G**
Exp 1 on B (d_f=$2d$)	0.753	0.164	0.093	0.103	0.442	0.089	0.123	0.124
Exp 2 on L (d_f=$4d$)	0.497	0.756	0.113	0.133	0.572	0.142	0.126	0.131
Exp 3 on G (d_f=$6d$)	0.116	0.147	0.704	0.128	0.127	0.153	0.210	0.175
Policy-Soup	**0.974**	**0.822**	**0.736**	**0.171**	**0.912**	**0.644**	**0.656**	**0.584**

B.2 Initialization Ablations

Expert initialization. Table 10 provides the condition-level results for the initialization ablation summarized in Table 5.

Table 10: **Full expert-initialization ablation.** RoboTwin 2.0 reports success rates averaged over shake bottle and place container plate, with 100 rollouts per task–configuration; PushT reports IoU. The last RoboTwin 2.0 mixed column includes held-out object texture, while the preceding six mix seen factors.

RoboTwin 2.0	**Seen Domains**				**Held-out**	**Mixed Domains**						
	L	**T**	**B**	**S**	**O**	**L+T**	**L+B**	**B+S**	**L+T +B**	**L+B +S**	**L+T +B+S**	**L+T+B +S+O**
Policy-Soup (w/o init)	0.72	0.46	0.51	0.77	0.78	0.47	0.55	0.53	0.20	0.22	0.20	0.17
Policy-Soup	**0.74**	**0.60**	**0.57**	**0.79**	**0.79**	**0.58**	**0.56**	**0.55**	**0.42**	**0.42**	**0.27**	**0.25**

PushT	**Seen Domains**			**Held-out**	**Mixed Domains**			
	B	**L**	**G**	**T**	**B+L**	**L+G**	**B+G**	**B+L+G**
Policy-Soup (w/o init)	0.970	0.807	0.702	0.164	0.884	0.482	0.643	0.457
Policy-Soup	**0.974**	**0.822**	**0.736**	**0.171**	**0.912**	**0.644**	**0.656**	**0.584**

B.3 Model Merging Comparison: Protocol and Results

For each task in Table 11, we merge the self- and cross-attention weights and biases stored in the corresponding consolidated checkpoint, aligning corresponding parameters across experts. All eight decoder layers are replaced, leaving one branch of each type per layer with mixture weight one, while non-attention parameters and normalization statistics are retained. The original POLICY-SOUP uses four frozen branches of each type and its learned output-aggregation weights. No further training follows replacement.

For corresponding expert attention parameters θ_i, uniform WA computes $\frac{1}{4}\sum_i \theta_i$, while TA computes $\sum_i \theta_i$ at scale 1.0. TA, TIES, and TSV use a zero reference, rather than a shared pretrained initialization, and scale 1.0. We apply the TIES operator Yadav et al. (2023b) to each expert's concatenated attention parameters across all layers, with target density 0.2, sign election, and disjoint-mean aggregation; threshold ties can retain more than the target density. We use TSV-M Gargiulo et al. (2025) with fused QKV matrices, retaining $\lfloor \min(m, n)/4 \rfloor$ singular directions per expert for an $m \times n$ matrix before orthogonalization, and averages one-dimensional biases. These merge settings are fixed rather than validation-tuned. The comparison covers L/T/B/S for both tasks, with 100 trials per method and condition, and reports unweighted condition means. Table 11 gives the full task-level results; Table 6 averages the two tasks equally for each condition, then averages the four conditions for Mean. All means and percentage-point differences are computed before rounding.

Table 11: **Comparison of attention merging strategies on RoboTwin 2.0.** Success rates (↑) use 100 trials per method and condition. L/T/B/S denote the `light`, `table`, `background`, and `sensor_noise` configurations. POLICY-SOUP retains four branches per attention type per decoder layer, whereas each merging alternative uses one; all non-attention components are unchanged, and no variant is retrained after replacement. Mean averages L/T/B/S within each task.

Method	L	T	B	S	Mean
(a) Shake bottle					
POLICY-SOUP (ours)	**0.92**	**0.89**	**0.62**	**0.97**	**0.850**
WA	0.47	0.32	0.42	0.31	0.380
TA	0.34	0.44	0.52	0.38	0.420
TIES	0.32	0.41	0.45	0.40	0.395
TSV	0.30	0.27	0.38	0.27	0.305
(b) Open laptop					
POLICY-SOUP (ours)	**0.56**	**0.50**	**0.54**	**0.70**	**0.575**
WA	0.14	0.20	0.16	0.15	0.163
TA	0.02	0.02	0.01	0.04	0.023
TIES	0.01	0.04	0.00	0.02	0.018
TSV	0.02	0.02	0.00	0.01	0.013

B.4 FIVE-EXPERT CONSOLIDATION

We evaluate five-expert consolidation on two additional RoboTwin 2.0 tasks, *grab roller* and *turn switch*. Unlike the main RoboTwin 2.0 setting, which trains on four atomic domains and holds out object texture, these experiments treat all five domains as seen. Table 12 reports the individual experts, POLICY-SOUP without expert initialization, and the full model.

On *grab roller*, POLICY-SOUP exceeds the strongest individual expert in three domains and matches it in the other two. On *turn switch*, it exceeds the strongest individual expert in all five domains, although several gains are small. Absolute success remains lower on *turn switch*, so consolidation does not remove the task's difficulty.

Expert initialization improves the reported success rate in every domain for both tasks. The largest full-model versus no-initialization gaps are 0.17 for *grab roller* and 0.24 for *turn switch*. These results support consolidation of five experts in the evaluated settings, without establishing scalability beyond this expert count.

Table 12: **Five-expert consolidation on two RoboTwin 2.0 tasks.** All numbers are success rates. L, T, B, S, and O denote lighting condition, table texture, background scene, sensor noise, and object texture, respectively; all five are seen domains. Underlined numbers indicate the best individual expert.

Grab Roller	**Seen Domains**				
	L	**T**	**B**	**S**	**O**
Expert 1	<u>0.86</u>	0.36	0.42	0.63	0.61
Expert 2	0.45	<u>0.72</u>	0.39	0.81	0.78
Expert 3	0.50	0.34	<u>0.69</u>	0.86	0.83
Expert 4	0.58	0.40	0.47	<u>0.91</u>	0.75
Expert 5	0.63	0.44	0.51	0.78	<u>0.95</u>
Policy-Soup (w/o init)	0.85	0.76	0.57	0.88	0.92
Policy-Soup	**0.89**	**0.77**	**0.74**	**0.91**	**0.95**

Turn Switch	**Seen Domains**				
	L	**T**	**B**	**S**	**O**
Expert 1	<u>0.29</u>	0.07	0.17	<u>0.38</u>	0.15
Expert 2	0.13	<u>0.27</u>	0.25	0.11	0.10
Expert 3	0.27	0.14	<u>0.31</u>	0.32	0.40
Expert 4	0.20	0.11	0.07	0.33	0.28
Expert 5	0.14	0.10	0.12	0.26	<u>0.41</u>
Policy-Soup (w/o init)	0.44	0.06	0.26	0.38	0.38
Policy-Soup	**0.46**	**0.30**	**0.32**	**0.40**	**0.42**