Mixture of Action Expert Embeddings: Multi-Task ACT

Suhyung Choi[1], **Youngseok Joo**[1], **Jun Ki Lee**[1 2], **Byoung-Tak Zhang**[1 2]

[1] Seoul National University
[2] AI Institute, Seoul National University
{s.choi, robinjoo1015, junkilee, btzhang}@snu.ac.kr

Abstract

Recent advances in imitation learning have enabled robotic policies with impressive performance. However, achieving general multi-task capabilities often requires models with large numbers of parameters and extensive datasets, resulting in computational inefficiency. This challenge is particularly pronounced in bimanual manipulation, where existing approaches are typically limited to single-task policies, necessitating separate models for each task and lacking generalization to multi-task scenarios. To address these challenges, we propose the Mixture of Action Expert Embeddings (MAE), a novel approach that facilitates a unified policy for multi-task bimanual manipulation without the need for large parameter models or additional task-specific datasets. By integrating MAE with the Action Chunking Transformer (ACT), our model achieves state-of-the-art performance on the ALOHA simulation benchmark, surpassing task-specific baselines on each task. Moreover, compared to the original ACT trained on multiple tasks, our MAE-ACT achieves a 67% success rate on challenging insertion tasks, whereas the original ACT achieves only a 17% success rate. We demonstrate that MAE-ACT effectively enables efficient multi-task learning and enhances the generalization capabilities of bimanual manipulation policies.

Introduction

The advancements in deep learning have extended beyond vision and natural language processing, showcasing significant potential in various domains such as reinforcement learning and robotics (LeCun, Bengio, and Hinton 2015; Silver et al. 2016; Zhao et al. 2023). Recently, imitation learning-based robotics has achieved substantial progress, demonstrating the feasibility of robotic systems learning complex tasks through human demonstrations (Brohan et al. 2022; Zitkovich et al. 2023; Zeng et al. 2023; Zhao et al. 2023; Collaboration et al. 2024). These approaches often rely on teleoperation to collect large datasets of demonstrations, enabling policies to imitate human actions effectively. However, achieving a generalizable robotic policy remains a formidable challenge. Previous works have demonstrated that unimanual manipulators are capable of executing a variety of tasks (Finn et al. 2017). Building upon this foundation, models with large parameters have been utilized to further enhance performance (Brohan et al. 2022; Zitkovich et al. 2023), trained on extensive datasets (Collaboration et al. 2024). Additionally, there have been efforts to create more general policies by integrating large language models (LLMs) (Kim et al. 2024; Ghosh et al. 2024). Despite these advancements, large foundational models exhibit significant variations in task success rates when tasks change slightly or when environmental conditions vary (Xie et al. 2024; Zhao et al. 2023). Moreover, they require enormous computational resources and massive datasets for training, making them impractical for widespread deployment.

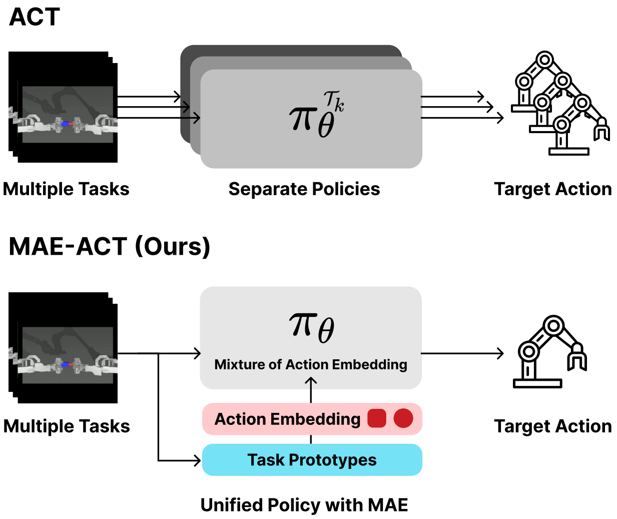

Figure 1: Comparison of ACT and MAE-ACT. ACT relies on separate policies for disjoint tasks, while MAE-ACT uses a unified policy that leverages both shared and task-specific representations through a mixture of action expert embeddings based on the task prototype.

Bimanual manipulation has emerged as a powerful approach in robotic systems, enabling tasks that go beyond the capabilities of unimanual setups (Zhao et al. 2023; Fu, Zhao, and Finn 2024; Fu et al. 2024; Shi et al. 2024; Team et al. 2024; Zhao et al. 2024). Notably, the Action Chunking Transformer (ACT) has demonstrated the feasibility of fine-grained, dexterous manipulation with bimanual robots, showcasing its potential for advancing complex robotic tasks (Zhao et al. 2023). ACT-based policies have been applied in diverse scenarios, such as tasks requiring additional mobility (Fu, Zhao, and Finn 2024), integration with humanoid hardware for human-like manipulation (Fu et al. 2024), executing natural language commands (Shi et al. 2024), and leveraging diffusion models for enhanced tabletop bimanual manipulation (Zhao et al. 2024). However, these approaches focus primarily on improving imitation of specific tasks rather than on generalization. As noted in (Zhao et al. 2024), current methods do not support multi-task learning within a single policy; instead, they require training separate policies for each task. This limitation arises because bimanual manipulation involves a larger action space and necessitates learning cooperative and more dexterous policies between two arms, making multi-task learning more challenging than in unimanual systems.

In this paper, we introduce the Mixture of Action Expert Embedding Action Chunking Transformer (MAE-ACT), a novel framework designed to enable bimanual manipulation systems to efficiently perform multiple tasks without requiring extensive parameters or reliance on large-scale external datasets. By incorporating a mixture of expert embeddings, our approach effectively generalizes across diverse tasks, addressing key scalability limitations of previous models. We demonstrate that MAE-ACT achieves state-of-the-art performance on a widely recognized benchmark for multi-task bimanual manipulation, representing a significant advancement toward practical and generalizable policies.

In summary, we make the following contributions:

- We propose a novel approach, the Mixture of Action Expert Embeddings (MAE), that enables multi-task capabilities without requiring additional expert networks, large policy models, or external datasets.
- By adapting task prototypes and action expert embeddings, we demonstrate that the policy effectively achieves task-specific objectives while leveraging shared representations, reaching state-of-the-art performance in the ALOHA simulation environment and proving its effectiveness in multi-task learning.
- Unlike large foundational models that lack explicit task identification and face challenges with task success, our approach uses a lightweight model that identifies tasks explicitly and dynamically mixes action expert embeddings, ensuring robust and efficient task execution while maintaining a compact model design.

Related Work

Multi-Task Models for Robotic Manipulation. Multi-tasking in robotic manipulation has emerged as a critical goal, enabling robots to perform diverse tasks under a unified framework (Rahmatizadeh et al. 2018; Gupta et al. 2021; Kalashnikov et al. 2021). Recent works in unimanual manipulation have introduced various multi-task policy learning approaches, including simple transformer architectures (Haldar, Peng, and Pinto 2024; Shridhar, Manuelli, and Fox 2022), semantic augmentation (Bharadhwaj et al. 2024), and diffusion-based techniques (Yan, Wu, and Wang 2024). Meanwhile, foundation models have redefined multi-task learning with transformer-based architectures (Brohan et al. 2022; Ghosh et al. 2024; Driess et al. 2023), large-scale multi-task robot datasets (Collaboration et al. 2024; Fang et al. 2023), and vision-language-action frameworks (Zitkovich et al. 2023; Kim et al. 2024). However, their success comes at the cost of extensive computational resources and a reliance on large, labeled datasets. Moreover, achieving remarkable results in unimanual manipulation, their extension to bimanual scenarios remains largely unexplored.

Models for Bimanual Robotic Manipulation. Bimanual manipulation offers enhanced dexterity and versatility by utilizing the broader action space enabled by two arms. The introduction of the Action Chunking Transformer (ACT) marked a pivotal advancement in bimanual manipulation (Zhao et al. 2023), predicting actions as chunk sequences through a noise-robust temporal ensemble. Subsequent works have extended this approach by integrating mobility capabilities (Fu, Zhao, and Finn 2024), enabling language-conditioned policy execution (Shi et al. 2024), adapting to humanoid hardware (Fu et al. 2024), and incorporating diffusion strategies (Zhao et al. 2024). Other efforts have explored extensions such as causal transformers (Zhang et al. 2024) and hierarchical attention mechanisms (Lee et al. 2024). Despite these advancements, most existing models for bimanual manipulation remain focused on single-task policies designed for specific manipulation scenarios. This approach requires separate policies for each task, which limits both generalization and scalability. In contrast, our model, MAE-ACT, seamlessly extends action chunking to enable multi-tasking without requiring additional external datasets or large parameter models.

Method

In this section, we present the Action Chunking Transformer (ACT) with Mixture of Action Expert Embeddings (MAE) and explain how it addresses multi-task challenges by leveraging task-specific and shared knowledge. A detailed architecture is summarized in Figure 2. We also provide an overview of the baseline model, ACT, describing its CVAE-based design and how it is extended with MAE to enable multi-tasking.

Action Chunking Transformer (ACT)

The Action Chunking Transformer (ACT) (Zhao et al. 2023) leverages a Conditional Variational Autoencoder (CVAE) (Sohn, Lee, and Yan 2015) architecture, consisting of a CVAE encoder and a CVAE decoder. The encoder generates a latent variable z, which the decoder uses to reconstruct an action sequence $\hat{a}_{t:t+c}$, where c is the chunk size of the action sequence and t is the timestep of the input. The model is trained using the following reconstruction loss, where $a_{t:t+c}$ represents the ground truth action sequence and $\hat{a}_{t:t+c}$ denotes the predicted actions over the same sequence:

$$\mathcal{L}_{\text{reconstruct}} = \left\| a_{t:t+c} - \hat{a}_{t:t+c} \right\|_1 . \quad (1)$$

During training, the encoder q_ϕ processes the input action sequence $a_{t:t+c}$ along with its condition, the current joint states j_t, to produce the latent variable z. This latent variable

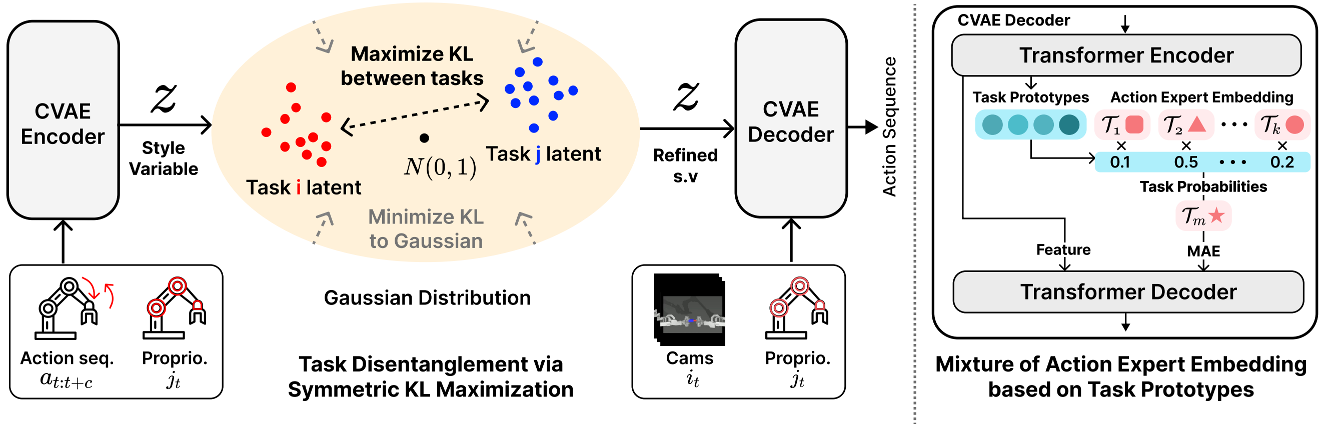

Figure 2: Overview of MAE-ACT. Our model utilizes the latent variable z to enforce disentanglement between tasks, and the mixture of action expert embeddings strengthen the joint task-specific representations by integrating task prototypes.

is optimized using the KL divergence regularization term:

$$\mathcal{L}_{\text{reg}} = D_{\text{KL}}(q_\phi(z|a_{t:t+c}, j_t) \parallel \mathcal{N}(0, I)). \tag{2}$$

The decoder takes the latent variable z along with its condition, the current observation o_t, which consists of multi-view camera images and joint states at timestep t. The latent variable z acts as a style variable, which ACT identifies as essential for capturing variations in human demonstrations in real-world scenarios. Using these inputs, the decoder reconstructs the action sequence $\hat{a}_{t:t+c}$. Consequently, the objective of ACT is to map the current observation o_t to the corresponding action $\hat{a}_{t:t+c}$. The overall training objective is defined as:

$$\theta^* = \arg\max_\theta \sum_{(o_t, a_{t:t+c}) \in \mathcal{D}} \log \pi_\theta \left(\hat{a}_{t:t+c} \mid o_t, z \right), \tag{3}$$

where π_θ represents the policy (decoder). During inference, the model relies solely on the decoder, with the latent variable initialized as $z = \mathbf{0}$.

Task Disentanglement in Latent Spaces

Learning effective multi-task representations requires balancing the sharing of common representations with the promotion of task-specific ones. However, sharing joint representations in latent spaces, particularly in Variational Autoencoders (VAEs), often leads to task interference, where overlapping latent representations hinder the performance of individual tasks (Ding et al. 2023; Lee and Pavlovic 2021; Xu et al. 2021). To address this challenge, we propose task disentanglement in the latent space of CVAE using a symmetric Kullback-Leibler (KL) divergence. This method encourages task-specific latent spaces to remain disentangled by maximizing the KL divergence between task-specific latent distributions during training. Since the KL divergence $D_{KL}(p \parallel q)$ is non-symmetric, we adopt a symmetric KL divergence, which computes the divergence in both directions for two distributions p and q and averages the results. Here, $p(\mathcal{Z})$ denotes the latent variable distribution produced by the encoder, equivalent to $q_\phi(z|a_{t:t+c}, j_t)$, the posterior distribution conditioned on the input action sequence and joint states. Formally, to maximize the symmetric KL divergence for all pairs of task-specific latent distributions $\mathcal{Z}_{\mathcal{T}_1}, \mathcal{Z}_{\mathcal{T}_2}, \ldots, \mathcal{Z}_{\mathcal{T}_k}$, where $\mathcal{T}_i$ represents the i-th task and k is the number of tasks, the loss function is defined as:

$$\begin{aligned} \mathcal{L}_{\text{symKL}} = -\frac{1}{k(k-1)} \sum_{i \neq j} \frac{1}{2} \Big(& D_{\text{KL}}\big(p(\mathcal{Z}_{\mathcal{T}_i}) \parallel p(\mathcal{Z}_{\mathcal{T}_j})\big) \\ & + D_{\text{KL}}\big(p(\mathcal{Z}_{\mathcal{T}_j}) \parallel p(\mathcal{Z}_{\mathcal{T}_i})\big)\Big), \end{aligned} \tag{4}$$

where the sum is computed over all possible pairs of task-specific latent distributions, promoting comprehensive disentanglement across tasks. Simultaneously, the task-specific latent representations are regularized to align closely with a Gaussian prior $\mathcal{N}(0, \mathbf{I})$, following the standard VAE objective and consistent with the KL regularization defined in Equation 2. This regularization is expressed as:

$$\mathcal{L}_{\text{KL-Gaussian}} = \sum_{i=1}^{k} D_{KL}\big(p(\mathcal{Z}_{\mathcal{T}_i}) \parallel \mathcal{N}(0, I)\big). \tag{5}$$

By minimizing $\mathcal{L}_{\text{KL-Gaussian}}$, we encourage the latent distributions of each task to remain bounded to Gaussian behavior. At the same time, minimizing $\mathcal{L}_{\text{symKL}}$ promotes separation and disentanglement of task-specific latent spaces. The interaction between these two forces introduces additional structure into the latent representations, reducing task interference and enhancing task-specific performance. While we hypothesize that this results in each task's latent distribution being bounded and well-structured, a formal proof of this property is left as future work. Instead, we focus on empirical evaluation to validate the proposed approach.

Mixture of Action Expert Embedding

We extend the ACT framework (Zhao et al. 2023) by introducing a novel mechanism that integrates task-specific and generalized representations into a unified framework. This section outlines the progression from the original ACT approach to the MAE-ACT framework, striking a balance between specialized and shared representations. The original

ACT policy is modeled using a CVAE architecture, where the decoder serves as the policy. It can be expressed as:

$$\pi_\theta^{\mathcal{T}_i}(\hat{a}_{t:t+c} \mid o_t^i, z^i), \tag{6}$$

where $\mathcal{T}_i$ denotes the i-th disjoint task, which also serves as its task ID when referencing task-specific representations, implying that ACT requires a separate policy for each task, which becomes inefficient in multi-task scenarios. To address this limitation, we propose a unified policy that incorporates task-specific representations given a task ID. This policy can be written as:

$$\pi_\theta(\hat{a}_{t:t+c} \mid o_t^i, z^i, \mathcal{T}_i), \tag{7}$$

However, in practical scenarios, the policy may not have access to the task ID and must instead infer the task. To enable this, we introduce a task identifier ψ, which maps the encoded feature to task probabilities. Using the task identifier, the task ID can be retrieved by taking the argmax of the predicted probabilities:

$$\mathcal{T}_i = \arg\max \psi(o_t^i, z^i). \tag{8}$$

The task identifier ψ, referred to as the *task prototype*, outputs a probability distribution over possible tasks based on the observation o_t^i and latent variable z^i. The task prototype is implemented as a linear layer followed by a softmax function, trained to classify the task. The CVAE decoder consists of a transformer encoder-decoder structure. The transformer encoder encodes features from o_t^i and z^i, while the transformer decoder utilizes these encoded features to generate actions. The task prototype takes the encoded features from the transformer encoder and predicts task probabilities. Denoting the encoder as f_{enc} and the decoder as f_{dec}, the policy becomes:

$$\pi_\theta(\hat{a}_{t:t+c} \mid o_t^i, z^i) = f_{\text{dec}}(f_{\text{enc}}(o_t^i, z^i), \psi(o_t^i, z^i)). \tag{9}$$

We enhance the policy's representational capabilities by introducing *action expert embeddings*, based on the task prototype. The transformer decoder takes as input an initialized output embedding of shape $\mathbb{R}^{C \times D}$, where C represents the number of queries (equivalent to the predicted action chunk size in ACT), and D is the hidden dimension. This output embedding is combined with a corresponding positional embedding of the same shape, $\mathbb{R}^{C \times D}$. To enhance task-specific adaptability, we replace traditional positional embeddings with task-specific embeddings, referred to as action expert embeddings. These action expert embeddings are element-wise added to the output embeddings. This allows the transformer decoder to effectively utilize task-specific representations. Let $\phi_i \in \mathbb{R}^{C \times 2D}$ denote the action expert embedding for task i, where the collection of all N task-specific embeddings is represented as $\phi \in \mathbb{R}^{N \times C \times 2D}$. Each task-specific embedding ϕ_i is transformed through a task-specific linear projection $W_i \in \mathbb{R}^{D \times 2D}$, mapping it into the decoder embedding space of shape $\mathbb{R}^{C \times D}$. The policy incorporating action expert embeddings with the inherited task prototype can be expressed as:

$$\pi_\theta(\hat{a}_{t:t+c} \mid o_t^i, z^i) = f_{\text{dec}}(f_{\text{enc}}(o_t^i, z^i), \phi_i), \tag{10}$$

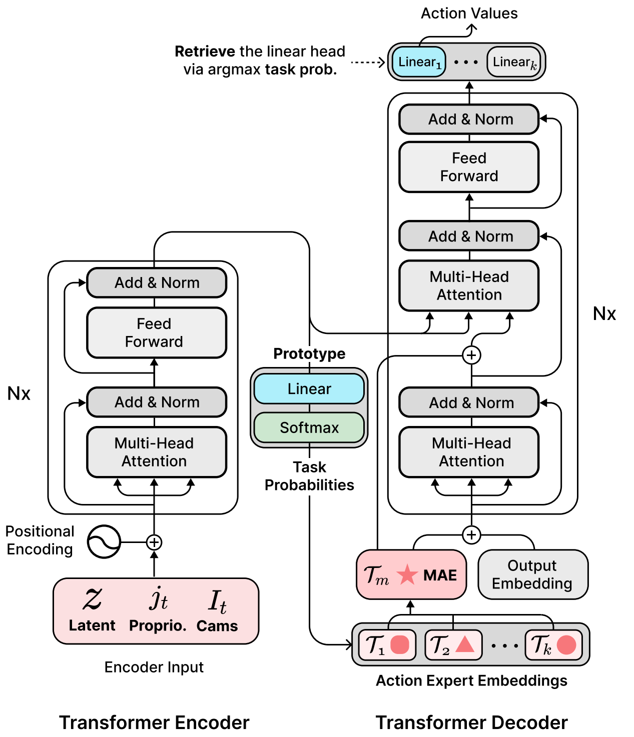

Figure 3: MAE-ACT policy with a transformer encoder and decoder. Task probabilities from the task prototype weight action expert embeddings, forming the Mixture of Action Expert Embeddings (MAE).

where:

$$\phi_i = \phi_{\arg\max \psi(o_t^i, z^i)}. \tag{11}$$

While the above formulation uses a single task-specific embedding, we extend the framework to incorporate a combination of action embeddings, referred to as the *mixture of action expert embeddings*. This approach enables the policy to leverage both task-specific representations and shared knowledge across tasks. To compute the mixture, we use the task probabilities ψ_i, which represent the probability of task i as predicted by the task prototype. These probabilities serve as weighting factors, determining the contribution of each task-specific embedding ϕ_i to the final mixed embedding. The resulting mixed action expert embedding is defined as:

$$\phi_{\text{mae}} = \sum_i \psi_i \phi_i. \tag{12}$$

The resulting policy using the mixed action expert embedding is expressed as:

$$\pi_\theta(\hat{a}_{t:t+c} \mid o_t^i, z^i) = f_{\text{dec}}(f_{\text{enc}}(o_t^i, z^i), \phi_{\text{mae}}). \tag{13}$$

By integrating task prototypes and action expert embeddings, the MAE-ACT framework enables efficient and generalizable multi-task learning, balancing task-specific and shared representations.

Task-Aware Action Head

While integrating ACT with the proposed MAE provides a sufficient level of generalization across tasks, we find that replacing the shared action head with task-specific linear heads further enhances performance. The selection of the task-specific linear head is retrieved directly without any heuristic by utilizing the task prototype as described in Equation 8, where the $\arg\max$ of the task probabilities determines the corresponding task. Denoting f'_{dec} as the decoder excluding the action head, the integration of the task-aware action head with the decoder can be expressed as:

$$\hat{a}_{t:t+c} = \text{Linear}_{\mathcal{T}_i}(f'_{\text{dec}}) \tag{14}$$

By leveraging the unified representation of the decoder with MAE, the addition of task-specific linear heads enables the seamless integration of task-specific adaptations with shared knowledge, thereby enhancing the framework's representational capacity and generalization capabilities. The final predicted action follows the static ACT setup (Zhao et al. 2023) and is expressed as a 14-degree-of-freedom vector, corresponding to the joint angles of the target positions.

Model Training

To train on multiple tasks, we adapt a joint training approach commonly used in multi-task learning (Caruana 1997; Navon et al. 2022; D'Eramo et al. 2024). Specifically, for a mini-batch sampled from each task dataset $(a^i_{t:t+c}, j^i_t, i^i_t) \sim D^i$, we forward each mini-batch through the same policy and compute the corresponding loss for each task. After obtaining the losses from all tasks, we sum them and perform backpropagation over the combined loss. To integrate task disentanglement in latent spaces, we use a latent disentanglement loss that combines the losses defined in Equations 4 and 5. The latent loss is formulated as:

$$\mathcal{L}_{\text{latent}} = \mathcal{L}_{\text{symKL}} + \mathcal{L}_{\text{KL-Gaussian}}. \tag{15}$$

For the task prototype and the mixture of action expert embeddings, we apply a cross-entropy loss for classifying the task ID given the encoded features from the Transformer encoder. The prototype loss is defined as:

$$\mathcal{L}_{\text{proto}} = -\sum_i p_i \log(\psi_i), \tag{16}$$

where p_i is the ground-truth task ID, and ψ_i represents the task probabilities obtained from the task prototype. For each task, the corresponding task-specific action expert embedding is backpropagated, while other task embeddings mixed with probabilities are detached during the gradient update. Finally, with the decoder action reconstruction loss Equation 1, the total loss for training is expressed as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{latent}} + \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{proto}}. \tag{17}$$

Each component is uniformly weighted during training.

Experiments

We evaluate MAE-ACT using the ALOHA simulation environment in ACT (Zhao et al. 2023), focusing on two simulated fine-manipulation tasks created in MuJoCo. Each task involves multi-stage fine manipulation, which is challenging as noted in prior works (Lee et al. 2024; Zhang et al. 2024; Zhao et al. 2023). We provide an overview of the tasks and baseline models, then compare success rates and rewards, showing that our model outperforms the baselines. Finally, we present ablation studies on model variations.

Simulated Tasks and Demonstration Datasets

We evaluate our approach on two challenging simulated tasks requiring fine-grained coordination and dexterous manipulation. The **Transfer Cube** task involves the right arm picking up a red cube and placing it into the left arm's gripper, with a narrow 1 cm clearance making the task prone to failure from minor misalignments. The **Peg Insertion** task requires the arms to pick up a socket and a peg, align them for insertion with a tight clearance of 5 mm, posing significant precision challenges. Object placements are randomized, with initial configurations uniformly distributed across 2D regions. We used the same 50-demonstration datasets from (Zhao et al. 2023), without any additional or external data for training.

Baselines

ACT (Zhao et al. 2023) is a representative model in bimanual manipulation, which is used as a backbone for many bimanual manipulation tasks. As mentioned above, ACT is based on a CVAE architecture, generating actions conditioned on its observations.
ARP (Zhang et al. 2024) is an autoregressive policy framework for robotic manipulation tasks. Using Chunking Causal Transformer (CCT), it predicts variable-sized action chunks for efficient sequence generation. ARP achieves state-of-the-art performance on RLBench (James et al. 2020) but focuses on generalizing within task variations rather than addressing multi-task learning.
Inter-ACT (Lee et al. 2024) is a imitation learning policy for bimanual manipulation. It employs hierarchical attention mechanisms and synchronization blocks to coordinate dual-arm actions, achieving state-of-the-art performance across bimanual tasks.

Evaluation on ALOHA Simulations

Metrics. We evaluate task performance using success rate and cumulative rewards. Each task is tested over 50 episodes across 15 random seeds. Tasks consist of four sequential stages, with each stage contributing 1 reward point. For example, in the peg insertion task, rewards are earned incrementally as the robot progresses through the following stages: touching the socket and peg, grasping both, aligning the peg with the socket, and successfully completing the insertion. A task is considered successful only when all stages are completed. Additionally, we compare the performance with and without *temporal ensemble*, a strategy adapted from the original ACT approach. Temporal ensemble involves performing inference at all timesteps and averaging the results to achieve a smoothing effect. In contrast, the non-temporal ensemble configuration performs inference from one chunk to next chunk, without ensemble.

(a) Peg Insertion

(b) Transfer Cube

Figure 4: ALOHA Simulation Environments. Figure 4a illustrates the Peg Insertion, where two arms align and insert a socket and a peg with a 5 mm clearance. Figure 4b depicts the Transfer Cube, requiring the right arm to transfer a red cube to the left arm with a 1 cm clearance. Both tasks demand precise dexterous manipulation under randomized object placements.

	Peg Insertion				Transfer Cube			
	w/ Temporal Ensemble		**w/o** Temporal Ensemble		**w/** Temporal Ensemble		**w/o** Temporal Ensemble	
Model	Success Rate	Reward	Success Rate	Reward	Success Rate	Reward	Success Rate	Reward
ACT (*Separate*)	0.54±0.11	387.30±10.38	0.54±0.11	365.23±18.46	0.97±0.03	**652.28**±20.36	**0.92**±0.04	**581.87**±27.23
ACT (*Unified*)	0.17±0.11	379.22±19.16	0.25±0.12	354.89±30.48	0.92±0.05	602.25±33.39	0.79±0.07	439.46±68.34
MAE-ACT (Ours)	**0.67**±0.10	**418.81**±15.87	**0.56**±0.09	**365.53**±12.72	**0.98**±0.02	650.11±42.65	0.83±0.07	537.15±44.79

Table 1: Comparative evaluation for two simulated tasks, with and without temporal ensemble. *Separate* refers to task-specific policies fully trained on single tasks, while *Unified* represents a single policy trained across multiple tasks. Bold values indicate the best performance, while underlined values highlight the failure of the original ACT to generalize across multiple tasks.

	Peg Insertion	**Transfer Cube**
Model	Success Rate	Success Rate
ACT (Zhao et al. 2023)	0.52	0.95
Inter-ACT (Lee et al. 2024)	0.44	0.82
ARP (Zhang et al. 2024)	0.24	0.94
MAE-ACT (Ours)	**0.67**	**0.98**

Table 2: Comparative evaluation of success rates across baselines. Bold values indicate the highest performance, while underlined values represent the second highest. Note that except MAE-ACT, other baselines are task-specific models fully optimized for individual tasks.

Quantitative Analysis of ACT and MAE-ACT

Table 1 summarizes the results for the Peg Insertion and Transfer Cube tasks, comparing the original ACT model with the proposed MAE-ACT framework. To evaluate MAE-ACT, which learns a unified policy across both tasks, we compare it with two variants of the original ACT: *Separate* ACT, where each task is trained independently, as in the original ACT, and *Unified* ACT, where both tasks are jointly trained using a single policy, similar to MAE-ACT.

Comparison with *Unified* ACT MAE-ACT demonstrates significant improvements in both generalization and task balance compared to *Unified* ACT. On Peg Insertion, MAE-ACT achieves a substantially higher success rate, while *Unified* ACT performs poorly and often fails the task. On the Transfer Cube task, MAE-ACT also outperforms *Unified* ACT, achieving consistently strong performance. These results reveal that *Unified* ACT overfits to the simpler Transfer Cube task, failing to generalize effectively to the more challenging Peg Insertion task. In contrast, MAE-ACT maintains robust and balanced performance across both tasks. Furthermore, MAE-ACT achieves higher rewards, which reflect average success rates across different task stages, indicating its superior ability to adapt and handle multi-task scenarios effectively.

Comparison with *Separate* ACT When compared to *Separate* ACT, which optimizes each task individually, MAE-ACT delivers competitive or superior performance on both tasks, even though it is trained jointly. For Peg Insertion, MAE-ACT achieves a higher success rate, while on Transfer Cube, MAE-ACT achieves comparable or slightly better performance depending on the configuration. In terms of reward, MAE-ACT matches or exceeds *Separate* ACT, demonstrating its ability to handle both tasks simultaneously without compromising performance. This indicates that MAE-ACT not only learns effective task-specific representations but also benefits from leveraging shared representations to improve overall results.

Quantitative Analysis of Baselines and MAE-ACT

We compare MAE-ACT to specialized baselines designed for single tasks, as shown in Table 2. While these baselines are optimized for predetermined tasks, MAE-ACT operates in a challenging multi-task setup, handling both Peg Insertion and Transfer Cube seamlessly. Despite this complexity, MAE-ACT consistently achieves the highest performance across both tasks. On Peg Insertion, MAE-ACT significantly outperforms baselines such as ARP (Zhang et al. 2024)

	Peg Insertion				Transfer Cube			
	w/ Temporal Ensemble		**w/o** Temporal Ensemble		**w/** Temporal Ensemble		**w/o** Temporal Ensemble	
Model	Success Rate	Reward	Success Rate	Reward	Success Rate	Reward	Success Rate	Reward
ACT (*Separate*)	0.54±0.11	387.3±10.38	0.54±0.11	365.23±18.46	0.97±0.03	**652.28**±20.36	**0.92**±0.04	**581.87**±27.23
ACT (*Unified*)	0.17±0.11	379.22±19.16	0.25±0.12	354.89±30.48	0.92±0.05	602.25±33.39	0.79±0.07	439.46±68.34
AE-ACT (*Unified*)	0.45±0.10	408.76±17.01	0.47±0.08	368.15±30.76	0.91±0.06	635.82±50.79	0.83±0.07	510.90±44.87
AE-ACT (*U*+*K*)	0.41±0.09	403.02±11.46	0.43±0.07	352.21±18.23	0.92±0.04	652.66±38.14	0.86±0.07	540.41±64.45
AE-ACT (*U*+*H*)	0.51±0.16	408.63±19.73	0.43±0.07	336.20±20.01	**0.99**±0.02	618.11±77.28	0.86±0.06	556.59±37.00
AE-ACT (*U*+*K*+*H*)	0.65±0.11	**427.35**±14.65	0.55±0.08	363.48±18.26	**0.99**±0.01	612.38±77.97	0.91±0.06	570.01±30.93
MAE-ACT (*Unified*)	0.46±0.13	411.88±9.16	0.50±0.08	369.85±18.40	0.89±0.07	626.13±60.70	0.80±0.11	533.12±59.88
MAE-ACT (*U*+*K*)	0.42±0.15	398.70±15.92	0.46±0.09	**371.35**±15.71	0.88±0.06	618.09±39.57	0.80±0.08	547.25±57.78
MAE-ACT (*U*+*H*)	0.66±0.11	413.88±19.29	0.51±0.10	365.17±20.51	0.97±0.03	644.47±51.50	0.81±0.08	526.96±55.94
MAE-ACT (*U*+*K*+*H*)	**0.67**±0.10	418.81±15.87	**0.56**±0.09	365.53±12.72	0.98±0.02	650.11±42.65	0.83±0.07	537.15±44.79

Table 3: Ablation of the components of MAE-ACT. AE refers to ACT with integrated Action Expert Embedding without mixture, while MAE represents ACT with Mixture of Action Expert Embedding. *U* indicates the Unified Policy, *K* denotes Latent Disentanglement with Symmetric KL Maximization, and *H* represents the task-aware Linear Head.

and Inter-ACT (Lee et al. 2024), achieving superior success rates and excelling in precise manipulation. On Transfer Cube, MAE-ACT also leads with the highest success rate, matching or exceeding task-specific baselines. These results demonstrate MAE-ACT's robust generalization and efficient balance between task-specific requirements and shared representation learning, enabling high performance across diverse tasks without interference.

Model	**Parameters (M)**
ACT	83.92
MAE-ACT	85.44

Table 4: Parameter Comparison Between ACT and MAE-ACT. With only a 1.8% increase in parameters, MAE-ACT demonstrated enhanced generalization across tasks.

Ablation Studies

We investigate the integration of each component within the MAE-ACT framework and assess its impact on overall performance. The ablation study highlights how incorporating the MAE-ACT components enhances the original ACT framework, enabling better generalization across tasks. Detailed results are presented in Table 3.

Action Expert Embedding. To evaluate the impact of Action Expert Embedding (AE), we conducted experiments by integrating task-specific action embeddings into ACT. The results, presented as AE in Table 3, also include configurations combining AE with latent disentanglement (*K*) and the task-aware linear head (*H*). In the absence of the mixture, action embeddings are selected using $\arg\max$ of the task prototype rather than through a probabilistic combination of embeddings from other tasks. Notably, simply adding task-specific action embeddings significantly improves performance on the Peg Insertion task compared to *Unified* ACT, while maintaining comparable performance on the Transfer Cube task. Although its performance is lower than *Separate* ACT, it still demonstrates a notable level of generalization across both tasks. These findings show that replacing positional embeddings with task-specific action expert embeddings enhances generalization and reduces overfitting to simpler tasks. Moreover, this result underscores the inherent representational capacity of the original ACT framework to effectively manage multiple tasks when augmented with appropriately designed embeddings.

Task-Aware Head. We further analyze the impact of the task-aware head on the overall framework. Task-specific heads are a commonly used strategy in multi-task learning (Crawshaw 2020). Here, we demonstrate how effectively they can generalize when integrated with bimanual manipulation policies. Specifically, we enable the task-specific head to be applied to each corresponding task by retrieving the task ID from the task prototype. As shown in Table 3, integrating the task-aware head consistently delivers significant performance gains compared to configurations without it, across all ablation combinations. By incorporating a task-specific linear head, the model successfully leverages task-specific representations built on shared representations. This design enables superior performance while avoiding interference between tasks, demonstrating the utility of this approach in multi-task setups.

Latent Disentanglement. Since the latent representation is initialized to zero during inference, latent disentanglement supports effective task conditioning rather than directly influencing the model. As shown in Table 3, adding latent disentanglement (*U*+*K*) to AE-ACT and MAE-ACT does not result in significant performance improvements compared to their *Unified* (*U*) counterparts. For Peg Insertion, incorporating disentanglement even slightly decreased performance, while for Transfer Cube, the performance remained largely unchanged. However, when latent disentanglement is combined with the task-aware head (*U*+*K*+*H*), both AE-ACT and MAE-ACT achieve better results on Peg Insertion and Transfer Cube tasks. This combination appears particularly beneficial, as the disentangled latent representation enhances the task-specific linear head's ability to decode and leverage task-specific features effectively. Notably, MAE-ACT (U+K+H) achieves a success rate of 0.67 on Peg Insertion and 0.98 on Transfer Cube, achieving state-of-the-art performance. Despite these promising results, the relatively high standard deviation indicates variability across trials. Further formal analysis is required to confirm these findings and fully understand the interplay between latent disentanglement and task-aware heads. We hypothesize that as the number of tasks increases, the benefits of disentanglement and task-specific heads will become more pronounced, offering clearer insights and advantages in future studies.

Mixture of Action Expert Embedding. The Mixture of Action Expert Embedding (MAE) facilitates leveraging joint representations across tasks while preserving task-specific representations. Although solely adapting the Action Expert Embedding (AE-$\{U\}$, $\{U$+$K\}$, $\{U$+$H\}$) appears sufficient for generalization, incorporating the mixture of action expert embedding (MAE-$\{U\}$, $\{U$+$K\}$, $\{U$+$H\}$) further enhances overall performance. When comparing AE and MAE with the integration of all components ($\{U$+K+$H\}$), the performance remains nearly identical, indicating that further investigation is required to fully understand and reveal its effects. We anticipate that including more tasks in future experiments would better highlight the advantages of using the mixture of action expert embedding.

Parameter Comparison between ACT and MAE-ACT. Table 4 highlights the parameter comparison between ACT and MAE-ACT. Key differences include the addition of a distinct action expert embedding with a dedicated linear layer for each task, the integration of task prototypes utilizing a single linear layer, and task-specific linear action heads. These architectural modifications lead to a modest parameter increase from 83.92M to 85.44M—an increment of only 1.8% compared to the original ACT. This demonstrates that MAE-ACT achieves efficient, well-generalized performance across tasks without necessitating a significant parameter expansion.

Limitations

Despite the improved performance of our MAE-ACT in bimanual manipulation compared to baselines, several limitations remain. First, our experiments are restricted to simulation environments, leaving its real-world applicability unproven. While ACT has shown strong performance in real-world settings, the potential of MAE-ACT for real-world multi-task scenarios remains promising but requires further validation beyond simulated conditions. Additionally, our evaluations are limited to the two tasks defined by the original ACT framework. This highlights the need for future research to explore and extend MAE-ACT to a broader range of tasks and applications.

Conclusion

In this paper, we introduce MAE-ACT, a novel approach that combines the Mixture of Action Expert Embeddings (MAE) with the Action Chunking Transformer (ACT) to address the complexities of multi-task bimanual manipulation. Our method enables a unified policy capable of handling multiple tasks without requiring large parameter models or extensive external datasets. Comprehensive experiments on the ALOHA simulation benchmark show that MAE-ACT outperforms single-task-specific models and the original ACT, achieving higher success rates. Although our work primarily focuses on simulations and a limited set of tasks, the promising results suggest that MAE-ACT has the potential to generalize to real-world applications. We believe that our method can extend bimanual manipulation to a variety of applications, such as language-conditioned tasks and mobile manipulation, enabling the exploration of more complex and diverse scenarios in robotic manipulation.

Acknowledgment

This work was partly supported by the IITP (RS-2021-II212068-AIHub /10%, RS-2021-II211343-GSAI /15%, RS-2022-II220951-LBA /15%, RS-2022-II220953-PICA /20%), NRF (RS-2024-00353991-SPARC /20%, RS-2023-00274280-HEI /10%), and KEIT (RS-2024-00423940 /10%) grant funded by the Korean government.

References

Bharadhwaj, H.; Vakil, J.; Sharma, M.; Gupta, A.; Tulsiani, S.; and Kumar, V. 2024. RoboAgent: Generalization and Efficiency in Robot Manipulation via Semantic Augmentations and Action Chunking. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 4788–4795.

Brohan, A.; Brown, N.; Carbajal, J.; Chebotar, Y.; Dabis, J.; Finn, C.; Gopalakrishnan, K.; Hausman, K.; Herzog, A.; Hsu, J.; et al. 2022. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*.

Caruana, R. 1997. Multitask learning. *Machine learning*, 28: 41–75.

Collaboration, E.; O'Neill, A.; Rehman, A.; Gupta, A.; Maddukuri, A.; Gupta, A.; Padalkar, A.; Lee, A.; Pooley, A.; Gupta, A.; Mandlekar, A.; Jain, A.; Tung, A.; Bewley, A.; Herzog, A.; Irpan, A.; Khazatsky, A.; Rai, A.; Gupta, A.; Wang, A.; Kolobov, A.; Singh, A.; Garg, A.; Kembhavi, A.; Xie, A.; Brohan, A.; Raffin, A.; Sharma, A.; Yavary, A.; Jain, A.; Balakrishna, A.; Wahid, A.; Burgess-Limerick, B.; Kim, B.; Schölkopf, B.; Wulfe, B.; Ichter, B.; Lu, C.; Xu, C.; Le, C.; Finn, C.; Wang, C.; Xu, C.; Chi, C.; Huang, C.; Chan, C.; Agia, C.; Pan, C.; Fu, C.; Devin, C.; Xu, D.; Morton, D.; Driess, D.; Chen, D.; Pathak, D.; Shah, D.; Büchler, D.; Jayaraman, D.; Kalashnikov, D.; Sadigh, D.; Johns, E.; Foster, E.; Liu, F.; Ceola, F.; Xia, F.; Zhao, F.; Frujeri, F. V.; Stulp, F.; Zhou, G.; Sukhatme, G. S.; Salhotra, G.; Yan, G.; Feng, G.; Schiavi, G.; Berseth, G.; Kahn, G.; Yang, G.; Wang, G.; Su, H.; Fang, H.-S.; Shi, H.; Bao, H.; Amor, H. B.; Christensen, H. I.; Furuta, H.; Bharadhwaj, H.; Walke, H.; Fang, H.; Ha, H.; Mordatch, I.; Radosavovic, I.; Leal, I.; Liang, J.; Abou-Chakra, J.; Kim, J.; Drake, J.; Peters, J.; Schneider, J.; Hsu, J.; Vakil, J.; Bohg, J.; Bingham, J.; Wu, J.; Gao, J.; Hu, J.; Wu, J.; Wu, J.; Sun, J.; Luo, J.; Gu, J.; Tan, J.; Oh, J.; Wu, J.; Lu, J.; Yang, J.; Malik, J.; Silvério, J.; Hejna, J.; Booher, J.; Tompson, J.; Yang, J.; Salvador, J.; Lim, J. J.; Han, J.; Wang, K.; Rao, K.; Pertsch, K.; Hausman, K.; Go, K.; Gopalakrishnan, K.; Goldberg, K.; Byrne, K.; Oslund, K.; Kawaharazuka, K.; Black, K.; Lin, K.; Zhang, K.; Ehsani, K.; Lekkala, K.; Ellis, K.; Rana, K.; Srinivasan, K.; Fang, K.; Singh, K. P.; Zeng, K.-H.; Hatch, K.; Hsu, K.; Itti, L.; Chen, L. Y.; Pinto, L.; Fei-Fei, L.; Tan, L.; Fan, L. J.; Ott, L.; Lee, L.; Weihs, L.; Chen, M.; Lepert, M.; Memmel, M.; Tomizuka, M.; Itkina, M.; Castro, M. G.; Spero, M.; Du, M.; Ahn, M.; Yip, M. C.; Zhang, M.; Ding, M.; Heo, M.; Srirama, M. K.; Sharma, M.; Kim, M. J.; Kanazawa, N.; Hansen, N.; Heess, N.; Joshi, N. J.; Suenderhauf, N.; Liu, N.; Palo, N. D.; Shafiullah, N. M. M.; Mees, O.; Kroemer, O.; Bastani, O.; Sanketi, P. R.; Miller, P. T.; Yin, P.; Wohlhart, P.; Xu, P.; Fagan, P. D.; Mitrano, P.; Sermanet, P.; Abbeel, P.; Sundaresan, P.; Chen, Q.; Vuong, Q.; Rafailov, R.; Tian, R.; Doshi, R.; Mart'in-Mart'in, R.; Baijal, R.; Scalise, R.; Hendrix, R.; Lin, R.; Qian, R.; Zhang, R.; Mendonca, R.; Shah, R.; Hoque, R.; Julian, R.; Bustamante, S.; Kirmani, S.; Levine, S.; Lin, S.; Moore, S.; Bahl, S.; Dass, S.; Sonawani, S.; Tulsiani, S.; Song, S.; Xu, S.; Haldar, S.; Karamcheti, S.; Adebola, S.; Guist, S.; Nasiriany, S.; Schaal, S.; Welker, S.; Tian, S.; Ramamoorthy, S.; Dasari, S.; Belkhale, S.; Park, S.; Nair, S.; Mirchandani, S.; Osa, T.; Gupta, T.; Harada, T.; Matsushima, T.; Xiao, T.; Kollar, T.; Yu, T.; Ding, T.; Davchev, T.; Zhao, T. Z.; Armstrong, T.; Darrell, T.; Chung, T.; Jain, V.; Kumar, V.; Vanhoucke, V.; Zhan, W.; Zhou, W.; Burgard, W.; Chen, X.; Chen, X.; Wang, X.; Zhu, X.; Geng, X.; Liu, X.; Liangwei, X.; Li, X.; Pang, Y.; Lu, Y.; Ma, Y. J.; Kim, Y.; Chebotar, Y.; Zhou, Y.; Zhu, Y.; Wu, Y.; Xu, Y.; Wang, Y.; Bisk, Y.; Dou, Y.; Cho, Y.; Lee, Y.; Cui, Y.; Cao, Y.; Wu, Y.-H.; Tang, Y.; Zhu, Y.; Zhang, Y.; Jiang, Y.; Li, Y.; Li, Y.; Iwasawa, Y.; Matsuo, Y.; Ma, Z.; Xu, Z.; Cui, Z. J.; Zhang, Z.; Fu, Z.; and Lin, Z. 2024. Open X-Embodiment: Robotic Learning Datasets and RT-X Models. arXiv:2310.08864.

Crawshaw, M. 2020. Multi-task learning with deep neural networks: A survey. *arXiv preprint arXiv:2009.09796*.

D'Eramo, C.; Tateo, D.; Bonarini, A.; Restelli, M.; and Peters, J. 2024. Sharing knowledge in multi-task deep reinforcement learning. *arXiv preprint arXiv:2401.09561*.

Ding, C.; Lu, Z.; Wang, S.; Cheng, R.; and Boddeti, V. N. 2023. Mitigating task interference in multi-task learning via explicit task routing with non-learnable primitives. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7756–7765.

Driess, D.; Xia, F.; Sajjadi, M. S. M.; Lynch, C.; Chowdhery, A.; Ichter, B.; Wahid, A.; Tompson, J.; Vuong, Q.; Yu, T.; Huang, W.; Chebotar, Y.; Sermanet, P.; Duckworth, D.; Levine, S.; Vanhoucke, V.; Hausman, K.; Toussaint, M.; Greff, K.; Zeng, A.; Mordatch, I.; and Florence, P. 2023. PaLM-E: An Embodied Multimodal Language Model. arXiv:2303.03378.

Fang, H.-S.; Fang, H.; Tang, Z.; Liu, J.; Wang, J.; Zhu, H.; and Lu, C. 2023. RH20T: A Robotic Dataset for Learning Diverse Skills in One-Shot. In *RSS 2023 Workshop on Learning for Task and Motion Planning*.

Finn, C.; Yu, T.; Zhang, T.; Abbeel, P.; and Levine, S. 2017. One-shot visual imitation learning via meta-learning. In *Conference on robot learning*, 357–368. PMLR.

Fu, Z.; Zhao, Q.; Wu, Q.; Wetzstein, G.; and Finn, C. 2024. HumanPlus: Humanoid Shadowing and Imitation from Humans. *arXiv preprint arXiv:2406.10454*.

Fu, Z.; Zhao, T. Z.; and Finn, C. 2024. Mobile ALOHA: Learning Bimanual Mobile Manipulation with Low-Cost Whole-Body Teleoperation. arXiv:2401.02117.

Ghosh, D.; Walke, H. R.; Pertsch, K.; Black, K.; Mees, O.; Dasari, S.; Hejna, J.; Kreiman, T.; Xu, C.; Luo, J.; Tan, Y. L.; Chen, L. Y.; Vuong, Q.; Xiao, T.; Sanketi, P. R.; Sadigh, D.; Finn, C.; and Levine, S. 2024. Octo: An Open-Source Generalist Robot Policy. In *Proceedings of Robotics: Science and Systems*. Delft, Netherlands.

Gupta, A.; Yu, J.; Zhao, T. Z.; Kumar, V.; Rovinsky, A.; Xu, K.; Devlin, T.; and Levine, S. 2021. Reset-Free Reinforcement Learning via Multi-Task Learning: Learning Dexterous Manipulation Behaviors without Human Intervention. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 6664–6671.

Haldar, S.; Peng, Z.; and Pinto, L. 2024. BAKU: An Efficient Transformer for Multi-Task Policy Learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

James, S.; Ma, Z.; Arrojo, D. R.; and Davison, A. J. 2020. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2): 3019–3026.

Kalashnikov, D.; Varley, J.; Chebotar, Y.; Swanson, B.; Jonschkowski, R.; Finn, C.; Levine, S.; and Hausman, K. 2021. MT-Opt: Continuous Multi-Task Robotic Reinforcement Learning at Scale. arXiv:2104.08212.

Kim, M. J.; Pertsch, K.; Karamcheti, S.; Xiao, T.; Balakrishna, A.; Nair, S.; Rafailov, R.; Foster, E.; Lam, G.; Sanketi, P.; Vuong, Q.; Kollar, T.; Burchfiel, B.; Tedrake, R.; Sadigh, D.; Levine, S.; Liang, P.; and Finn, C. 2024. OpenVLA: An Open-Source Vision-Language-Action Model. arXiv:2406.09246.

LeCun, Y.; Bengio, Y.; and Hinton, G. 2015. Deep learning. *nature*, 521(7553): 436–444.

Lee, A.; Chuang, I.; Chen, L.-Y.; and Soltani, I. 2024. InterACT: Inter-dependency Aware Action Chunking with Hierarchical Attention Transformers for Bimanual Manipulation. *arXiv preprint arXiv:2409.07914*.

Lee, M.; and Pavlovic, V. 2021. Private-shared disentangled multimodal vae for learning of latent representations. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 1692–1700.

Navon, A.; Shamsian, A.; Achituve, I.; Maron, H.; Kawaguchi, K.; Chechik, G.; and Fetaya, E. 2022. Multi-Task Learning as a Bargaining Game. In Chaudhuri, K.; Jegelka, S.; Song, L.; Szepesvari, C.; Niu, G.; and Sabato, S., eds., *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, 16428–16446. PMLR.

Rahmatizadeh, R.; Abolghasemi, P.; Bölöni, L.; and Levine, S. 2018. Vision-Based Multi-Task Manipulation for Inexpensive Robots Using End-to-End Learning from Demonstration. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 3758–3765.

Shi, L. X.; Hu, Z.; Zhao, T. Z.; Sharma, A.; Pertsch, K.; Luo, J.; Levine, S.; and Finn, C. 2024. Yell At Your Robot: Improving On-the-Fly from Language Corrections. volume 20. ISBN 9798990284807.

Shridhar, M.; Manuelli, L.; and Fox, D. 2022. Perceiver-Actor: A Multi-Task Transformer for Robotic Manipulation. In *6th Annual Conference on Robot Learning*.

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. 2016. Mastering the game of Go with deep neural networks and tree search. *nature*, 529(7587): 484–489.

Sohn, K.; Lee, H.; and Yan, X. 2015. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28.

Team, A. .; Aldaco, J.; Armstrong, T.; Baruch, R.; Bingham, J.; Chan, S.; Draper, K.; Dwibedi, D.; Finn, C.; Florence, P.; Goodrich, S.; Gramlich, W.; Hage, T.; Herzog, A.; Hoech, J.; Nguyen, T.; Storz, I.; Tabanpour, B.; Takayama, L.; Tompson, J.; Wahid, A.; Wahrburg, T.; Xu, S.; Yaroshenko, S.; Zakka, K.; and Zhao, T. Z. 2024. ALOHA 2: An Enhanced Low-Cost Hardware for Bimanual Teleoperation. arXiv:2405.02292.

Xie, A.; Lee, L.; Xiao, T.; and Finn, C. 2024. Decomposing the generalization gap in imitation learning for visual robotic manipulation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 3153–3160. IEEE.

Xu, J.; Ren, Y.; Tang, H.; Pu, X.; Zhu, X.; Zeng, M.; and He, L. 2021. Multi-VAE: Learning disentangled view-common and view-peculiar visual representations for multi-view clustering. In *Proceedings of the IEEE/CVF international conference on computer vision*, 9234–9243.

Yan, G.; Wu, Y.-H.; and Wang, X. 2024. DNAct: Diffusion Guided Multi-Task 3D Policy Learning. arXiv:2403.04115.

Zeng, F.; Gan, W.; Wang, Y.; Liu, N.; and Yu, P. S. 2023. Large language models for robotics: A survey. *arXiv preprint arXiv:2311.07226*.

Zhang, X.; Liu, Y.; Chang, H.; Schramm, L.; and Boularias, A. 2024. Autoregressive action sequence learning for robotic manipulation. *arXiv preprint arXiv:2410.03132*.

Zhao, T. Z.; Kumar, V.; Levine, S.; and Finn, C. 2023. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. volume 19. ISBN 978-0-9923747-9-2.

Zhao, T. Z.; Tompson, J.; Driess, D.; Florence, P.; Ghasemipour, K.; Finn, C.; and Wahid, A. 2024. ALOHA Unleashed: A Simple Recipe for Robot Dexterity. arXiv:2410.13126.

Zitkovich, B.; Yu, T.; Xu, S.; Xu, P.; Xiao, T.; Xia, F.; Wu, J.; Wohlhart, P.; Welker, S.; Wahid, A.; Vuong, Q.; Vanhoucke, V.; Tran, H.; Soricut, R.; Singh, A.; Singh, J.; Sermanet, P.; Sanketi, P. R.; Salazar, G.; Ryoo, M. S.; Reymann, K.; Rao, K.; Pertsch, K.; Mordatch, I.; Michalewski, H.; Lu, Y.; Levine, S.; Lee, L.; Lee, T.-W. E.; Leal, I.; Kuang, Y.; Kalashnikov, D.; Julian, R.; Joshi, N. J.; Irpan, A.; Ichter, B.; Hsu, J.; Herzog, A.; Hausman, K.; Gopalakrishnan, K.; Fu, C.; Florence, P.; Finn, C.; Dubey, K. A.; Driess, D.; Ding, T.; Choromanski, K. M.; Chen, X.; Chebotar, Y.; Carbajal, J.; Brown, N.; Brohan, A.; Arenas, M. G.; and Han, K. 2023. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control. In Tan, J.; Toussaint, M.; and Darvish, K., eds., *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, 2165–2183. PMLR.