BAKU: An Efficient Transformer for Multi-Task Policy Learning

Siddhant Haldar **Zhuoran Peng** **Lerrel Pinto**

New York University

Abstract

Training generalist agents capable of solving diverse tasks is challenging, often requiring large datasets of expert demonstrations. This is particularly problematic in robotics, where each data point requires physical execution of actions in the real world. Thus, there is a pressing need for architectures that can effectively leverage the available training data. In this work, we present BAKU, a simple transformer architecture that enables efficient learning of multi-task robot policies. BAKU builds upon recent advancements in offline imitation learning and meticulously combines observation trunks, action chunking, multi-sensory observations, and action heads to substantially improve upon prior work. Our experiments on 129 simulated tasks across LIBERO, Meta-World suite, and the Deepmind Control suite exhibit an overall 18% absolute improvement over RT-1 and MT-ACT, with a 36% improvement on the harder LIBERO benchmark. On 30 real-world manipulation tasks, given an average of just 17 demonstrations per task, BAKU achieves a 91% success rate. Videos of the robot are best viewed at baku-robot.github.io.

1 Introduction

Learning generalist policies that can solve multiple tasks is a long standing problem in decision making and robotics. While significant advances have been made in computer vision [4, 59] and natural language processing [2, 53, 69], algorithms that can effectively do so for physical agents are far behind. A key reason for this is the scale of available data. While large-scale datasets in vision and language can readily be amassed from the Internet, robotics presents a unique challenge. Given its interactive nature, data acquisition requires physical engagement with the world, making robot data considerably more laborious to obtain in terms of both time and financial costs.

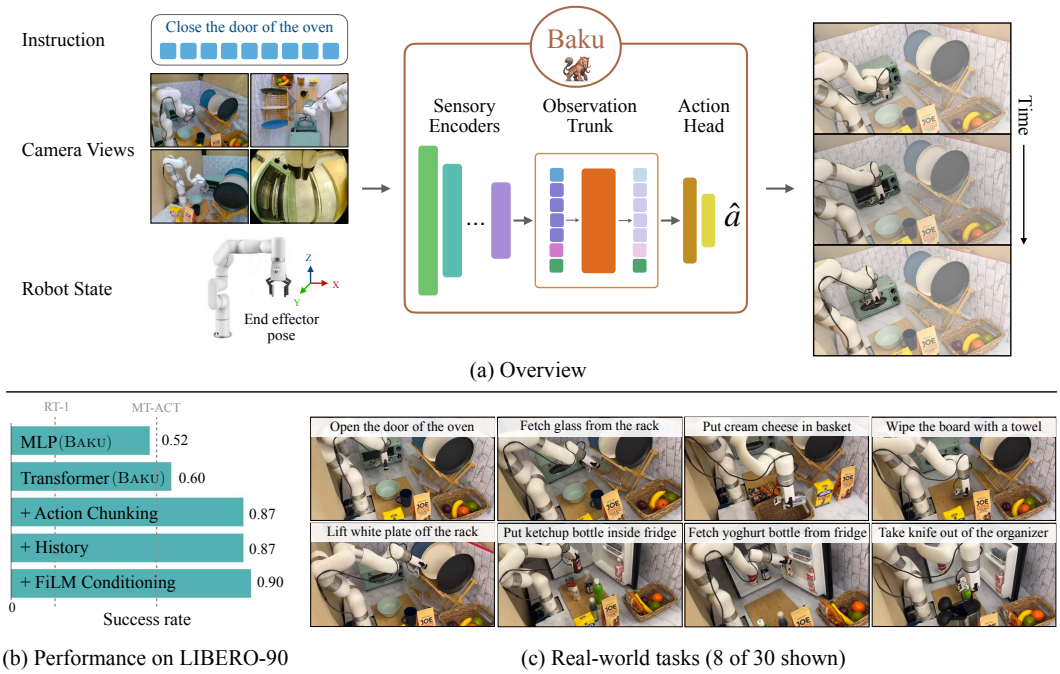
A prominent approach for training multi-task policies is to bite the bullet and collect large amounts of data, often by contracting teleoperators [37, 6, 61]. However, policies trained on such data are quite inefficient, often achieving performance far below independently trained single-task policies [75, 44, 57]. The current best answer to solve this problem is unfortunately to collect even more demonstration data from experts.

In this work, we present BAKU, a simple architecture for multi-task policy learning that provides highly efficient training, particularly in data-scarce problems such as robotics. BAKU builds upon recent work in multitask learning [5, 6] and has three key features. First, a transformer encoder that fuses information from multiple modalities like vision and language while incorporating temporal context. Second, a FiLM-conditioned [46] visual encoder helps the model learn task-specific representations by adapting the encoder to the task. Third, an action prediction head that is separated from the observational encoding trunk, enabling BAKU to be easily retrofitted with state-of-the-art action

Correspondence to: siddhanthaldar@nyu.edu

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

 https://doi.org/10.52202/079017-4484



(a) Overview

(b) Performance on LIBERO-90

(c) Real-world tasks (8 of 30 shown)

Figure 1: **(a)** We present BAKU, a simple transformer architecture learning multi-task policies across a diverse range of tasks. BAKU encodes inputs from different modalities using modality-specific encoders. The encoded representations are merged in an observation trunk before predicting actions through an action head. **(b)** We develop a unified policy for 90 tasks in the LIBERO-90 benchmark, discussing design choices that impact multi-task performance. **(c)** On our xArm robot, BAKU can learn a single multi-task policy for 30 tasks with an average of 17 demonstrations collected per task.

generation models [39, 60, 31, 10]. The novelty of BAKU hence lies in carefully combining these ideas to produce a new transformer architecture particularly suited for multitask decision making.

To demonstrate the effectiveness of BAKU, we run extensive experiments on 129 simulated tasks across LIBERO [34], Meta-World [76], and DeepMind Control [67], and 30 robotic manipulation tasks on an xArm robot (see Fig. 1). Our main findings are summarized below:

1. BAKU exhibits an overall 18% absolute performance improvement over prior state-of-the-art multi-task learning algorithms on 129 tasks across 3 simulated environment suites (Section 4.1). BAKU sets a state-of-the-art performance on LIBERO with 90% average success rate, a 36% absolute improvement over prior work (Table 1).
2. On real-world tasks, with an average of 17 demonstrations per task, BAKU achieves an average success rate of 91% across 30 diverse tasks in a multi-task kitchen environment, with randomized object initialization. This outperforms prior state-of-the-art algorithms by 35% (Section 4.2).
3. Through an ablation analysis, we study the importance of each component in BAKU (Section 4.4), particularly the role of action chunking [77] and a multimodal action head in boosting performance, especially in our real-world experiments.

All of our datasets, and training and evaluation code will be made publicly available. Videos of our trained policies can be seen here: baku-robot.github.io.

2 Background

Imitation Learning: The goal of imitation learning is to learn a behavior policy π^b given access to either the expert policy π^e or trajectories derived from the expert policy $\mathcal{T}^e$. While there are a multitude of settings with differing levels of access to the expert [68], this work operates in the setting where the agent only has access to observation-based trajectories, i.e. $\mathcal{T}^e \equiv \{(o_t, a_t)_{t=0}^T\}_{n=0}^N$. Here

https://doi.org/10.52202/079017-4484

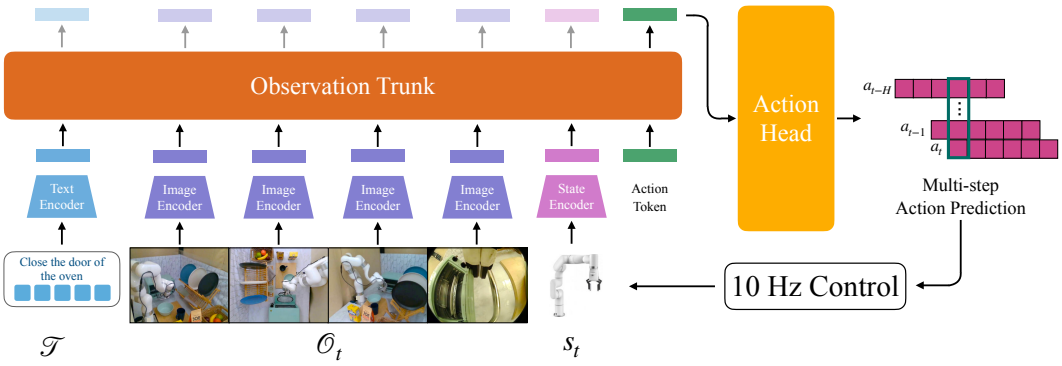


Figure 2: Overview of BAKU, broken down into modality-specific sensory encoders, an observation trunk, and an action head predicting a chunk of actions. BAKU takes as input observations from multiple camera views $\mathcal{O}_t$, robot proprioceptive state s_t and a task instruction $\mathcal{T}$ and enables performing closed-loop control at 10Hz in our real world experiments on the xArm.

N and T denote the number of trajectory rollouts and episode timesteps respectively. We choose this specific setting since obtaining observations and actions from expert or near-expert demonstrators is feasible in real-world settings [77, 24] and falls in line with recent work in this area [77, 11, 31, 10].

Multi-task Behavior Cloning: Behavior Cloning (BC) corresponds to solving the maximum likelihood problem shown in Eq. 1. Here $\mathcal{T}^e$ refers to expert demonstrations. When parameterized by a normal distribution with fixed variance, the objective can be framed as a regression problem where, given observations o^e, π^{BC} needs to output a^e.

$$\mathcal{L}^{BC} = \mathbb{E}_{(o^e,a^e)\sim\mathcal{T}^e} \|a^e - \pi^{BC}(o^e)\|^2 \tag{1}$$

After training, it enables π^{BC} to mimic the actions corresponding to the observations seen in the demonstrations. In multi-task settings, we use the same formulation for BC but condition the action prediction on a goal variable g^e. Thus, the loss function for multi-task BC becomes the following.

$$\mathcal{L}^{BC} = \mathbb{E}_{(o^e,a^e,g^e)\sim\mathcal{T}^e} \|a^e - \pi^{BC}(o^e|g^e)\|^2 \tag{2}$$

In this work, we represent goals as either a text description of the task [6, 5] or a goal image [11, 72].

3 BAKU

The design of multi-task learning algorithms involves numerous decisions regarding model architecture and component selection. This often results in complex architectures where the importance of individual components is sometimes unclear. In this work, we perform a systematic and thorough ablation study across the various multi-task learning architectures proposed by prior works [6, 5, 63] and introduce BAKU, a simple architecture for multi-task policy learning. To facilitate our analysis, we divide the overall model architecture into three main components: sensory encoders, an observation trunk, and an action head. Sensory encoders process raw sensor inputs from different modalities into useful feature representations. The observation trunk combines the encoded information from the different modalities. Finally, the action head utilizes the combined information to predict actions. Below, we describe these three components in detail, with additional algorithmic details provided in Appendix A.

3.1 Sensory Encoders

In the real-world, robots encounter diverse data modalities, including vision, depth feedback, proprioceptive feedback, and task instructions in various forms such as text, goal images, or task videos. In BAKU, we focus on vision, robot proprioception, and text or goal image based task instructions. For vision, we use a ResNet-18 [19] visual encoder to process images of the scene, enhanced with

https://doi.org/10.52202/079017-4484

a FiLM [46] layer to integrate task-specific information. Robot proprioception data is processed through a two-layer multilayer perception (MLP) encoder. For text, we use a 6-layer version of MiniLM [73] provided in Sentence Transformers [54]. We project the representations obtained from all modalities to the same dimensionality through additional MLP layers, to facilitate combining the encoded information. We have included a description of FiLM conditioning in Appendix A.1.

3.2 Observation Trunk

The encoded inputs from all sensory modalities are combined in the observation trunk. We explore two variants of the trunk network:

Multilayer Perceptron (MLP) The encoded inputs are concatenated into a single feature vector and passed through a multilayer perceptron. When using a history of observations, the inputs corresponding to all time steps are concatenated.

Transformer Each encoded input is treated as an observation token and passed through a transformer decoder network [70]. A learnable action token is appended to the list of observation tokens and used to predict the action. When using a historical observation, a separate action token is added for each time step to enable predicting actions for all time steps. A causal mask is applied to the transformer to ensure that actions are predicted solely based on past observations.

Both variants output action feature vectors (corresponding to the action tokens for a transformer), which are then passed through an action head to predict actions.

3.3 Action Head

The final component of our architecture is the action head, an action prediction module that takes as input the action feature vectors obtained from the observation trunk and predicts the corresponding actions. An independent action prediction module enables us to unify several state-of-the-art action generation models within the same framework. We experiment with five action head variants: vanilla MLP, Gaussian Mixture Model (GMM) [36], Behavior Transformer (BeT) [60], Vector-Quantized Behavior Transformer (VQ-BeT) [31], and diffusion policy [45, 10, 55]. Considering the temporal correlation in robot movements, we follow prior work [77, 5] and include action chunking with exponential temporal averaging to produce smoother behaviors and counteract the covariate shift often seen in low-data imitation learning scenarios. In contrast to previous works [77, 5] that decode actions for each time step separately, we predict the action chunk as a single concatenated vector. We find that this simplification improves performance (see Table 1). More details about each action head variant has been provided in Appendix A.2 along with details about the exponential temporal smoothing technique in Appendix A.3.

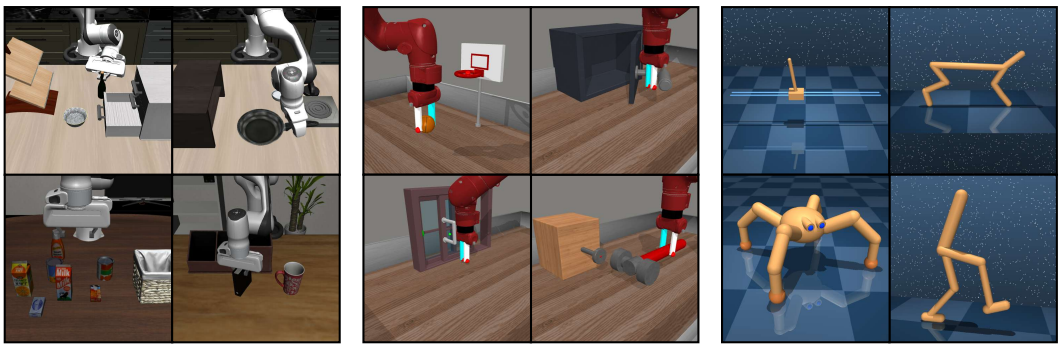
3.4 Putting it all together

Our proposed architecture is depicted in Figure 2. Through extensive experimentation (see Section 4), our final architecture includes a modified FiLM-conditioned ResNet-18 vision encoder (provided with the LIBERO benchmark [34]), an MLP encoder for robot proprioception, and a pre-trained text encoder for task instructions. For environments with multiple camera views, we use a common visual encoder across all views. We provide only the current observation as an input and the observation trunk uses a causal transformer decoder architecture [29]. The base version of our model uses an MLP action head with action chunking and temporal smoothing to produce smoother motions. We also experiment with multimodal variants of action heads and have provided the results in Section 4.4.

The parameter counts are approximately 2.1M for the sensory encoders, 6.5M for the observation trunk, and 1.4M for the action head, bringing the total model size to approximately 10M parameters.

4 Experiments

Our experiments are designed to answer the following questions: (a) How well does BAKU work for multi-task learning? (b) How does BAKU perform on real-world tasks? (c) How does BAKU perform



(a) LIBERO-90 (b) Metaworld (C) DM Control

Figure 3: BAKU is evaluated on 3 simulated benchmarks - LIBERO, Meta-World, and DM Control.

on long-horizon tasks? (d) What design decisions affect multi-task policy learning? Additional results and analysis have been provided in Appendix D.

Simulation Tasks: We experiment with 90 manipulation tasks from the LIBERO-90 benchmark [34], 30 manipulation tasks from Meta-World suite [76], and 9 locomotion tasks from DeepMind Control Suite (DMC) [67]. Figure 3 depicts the simulated environments. For LIBERO-90, we use 50 demonstrations per task provided with the benchmark and use images from third-person and gripper camera views, as well as robot proprioception as input. For Meta-World, we obtain 35 demonstrations per task from an expert policy trained with demonstration-guided reinforcement learning [17, 18], using only the third-person view as input. We use images of size 128×128 for LIBERO-90 and 84×84 for Meta-World. For DMC, we train state-based locomotion policies using 500 demonstrations per task obtained from experts trained with DrQ-v2 [74]. All evaluations are conducted using 10 policy rollouts per task. More details about the simulated environments can be found in Appendix B.

Robot Tasks: Our real-world experiments are performed on a Ufactory xArm 7 robot with an xArm Gripper in a multi-task kitchen environment. The policies are trained on RGB images of size 128×128 obtained from four different camera views, including an egocentric camera attached to the robot gripper. The action space comprises the robot end effector pose and the gripper state. We collect a total of 520 demonstrations across 30 tasks, averaging 17 demonstrations per task. The demonstrations were collected using a VR-based teleoperation system [24] at a 30Hz frequency. The learned policies are deployed at 10Hz. Figure 4 shows selected tasks from our real-world environment. More details about each task and robot control can be found in Appendix C.

Strong Baselines: In this section, we provide a detailed explanation of our baselines in relation to BAKU.

1. **MT-ACT [5]:** Multi-task Action-Chunking Transformer (MT-ACT) is a state-of-the-art transformer encoder-decoder architecture for learning multi-task policies. MT-ACT extends Action-Chunking Transformer (ACT) [77] to a multi-task setting. MT-ACT takes as input observations from multiple camera views, robot proprioception, and task instructions. Each input modality passes through dedicated encoders. The encoded observations are then fused in a transformer encoder, the output of which conditions a transformer decoder to predict chunks of future actions. Each predicted action corresponds to a position embedding input to the decoder. In ACT and MT-ACT, a conditional variational autoencoder (CVAE) is used to learn a multimodal style variable which conditions the encoder to deal with multimodal action distributions. During inference, the style variable is set to zero, leading to unimodal behavior. In contrast, BAKU uses a decoder-only transformer architecture that directly predicts action features corresponding to past observations. This enables us to (1) leverage recent advances in multimodal action generation by plugging in several unimodal and multimodal heads for action prediction, and (2) incorporate a history of observations to predict actions for each time step in the history (see Section 4.4 for results on the

https://doi.org/10.52202/079017-4484

Table 1: Performance of multi-task policies learned using BAKU on 3 simulated benchmarks - LIBERO-90, Meta-World, and DM Control - and a real xArm robot. We observe that BAKU significantly outperforms prior work on both simulated and real world tasks.

Method	LIBERO-90 (90 tasks)	Meta-World (30 tasks)	DMC (9 tasks)	Real Robot (20 tasks)
RT-1	0.16	0.65	0.66	0.37
MTACT	0.54	0.13	0.59	0.56
BAKU **(Ours)**	**0.9**	**0.79**	**0.7**	0.86
BAKU w/ VQ-BeT **(Ours)**	**0.9**	0.78	**0.7**	**0.91**

use of observation history). Further, using a multimodal action head enables BAKU to exhibit multimodal behavior during inference, improving real-world performance (Table 1).

2. **RT-1 [6]:** RT-1 is a transformer-based multi-task policy learning architecture that models actions as discrete classes by uniformly discretizing them into bins. RT-1 uses a FiLM-conditioned vision encoder (ResNet-18 in our implementation), but instead of directly using the final 512-dimensional representation, it splits an intermediate feature map of size $k \times k \times 512$ into k^2 tokens of 512 dimensions each. These tokens are passed through a Token Learner [58] module to reduce them to 8 tokens per image. The reduced number of tokens is then passed through a decoder-only transformer architecture to predict a discrete action. In contrast, BAKU directly uses the final 512-dimensional representation from the vision encoder, without summarizing tokens via a token learner. Additionally, BAKU predicts a continuous action through an unimodal or multimodal action head. Based on our experiments (Table 1), we observe that these design choices in BAKU lead to significant improvements in performance over RT-1.

4.1 How well does BAKU work for multi-task learning?

We evaluate the multi-task performance of BAKU on 90 tasks from the LIBERO-90 benchmark, 30 tasks from Meta-World, and 9 tasks from DMC. Table 1 compares the performance of BAKU with our baselines, RT-1 [6] and MT-ACT [5]. BAKU outperforms the strongest baseline by 36% and 14% on LIBERO-90 and Meta-World respectively, demonstrating more effective multi-task learning on complex manipulation tasks. On the simpler DMC locomotion tasks, BAKU outperforms the strongest baseline by 4%. Overall, these results suggest that BAKU more effectively leverages relationships between tasks to achieve superior multi-task learning performance compared to prior methods.

4.2 How does BAKU perform on real-world tasks?

We evaluate BAKU on 30 manipulation tasks in our real-world kitchen environment, comparing it with MT-ACT and RT-1. During evaluations, the xArm was always initialized at the same pose and the objects being manipulated were placed in a fixed set of positions for all methods. We conducted 5 evaluation runs per task, totaling 150 evaluation runs per method. Table 1 includes our real-world results. We observe that BAKU achieves an 86% success rate across all tasks, outperforming the strongest baseline by 30%. Replacing the MLP action head with a multimodal VQ-BeT [31] head further improves the success rate to 91%, outperforming the strongest baseline by 35%. Figure 4 shows real-world rollout trajectories for a selected task set with all tasks included in Appendix C. Appendix D.1 provides the task-wise performance for each method. Overall, these results indicate BAKU's promise for deploying multi-task policies on real-world robotic systems.

4.3 How does BAKU perform on long-horizon tasks?

We also evaluate BAKU on long-horizon tasks in the simulated LIBERO-10 benchmark and our real-world multi-task kitchen environment. Table 2 provides the results on 10 tasks in LIBERO-10 and 5 long-horizon tasks in the real kitchen environment, each composed of two shorter tasks chained sequentially. We use 50 demonstrations per task for LIBERO-10 and an average of 19 demonstrations per task for the real robot. We observe that BAKU significantly outperforms our strongest baseline, MT-ACT, on these long horizon tasks, achieving on average 19% higher success rate. This highlights BAKU's ability to learn policies that can effectively plan and execute sequences of actions over

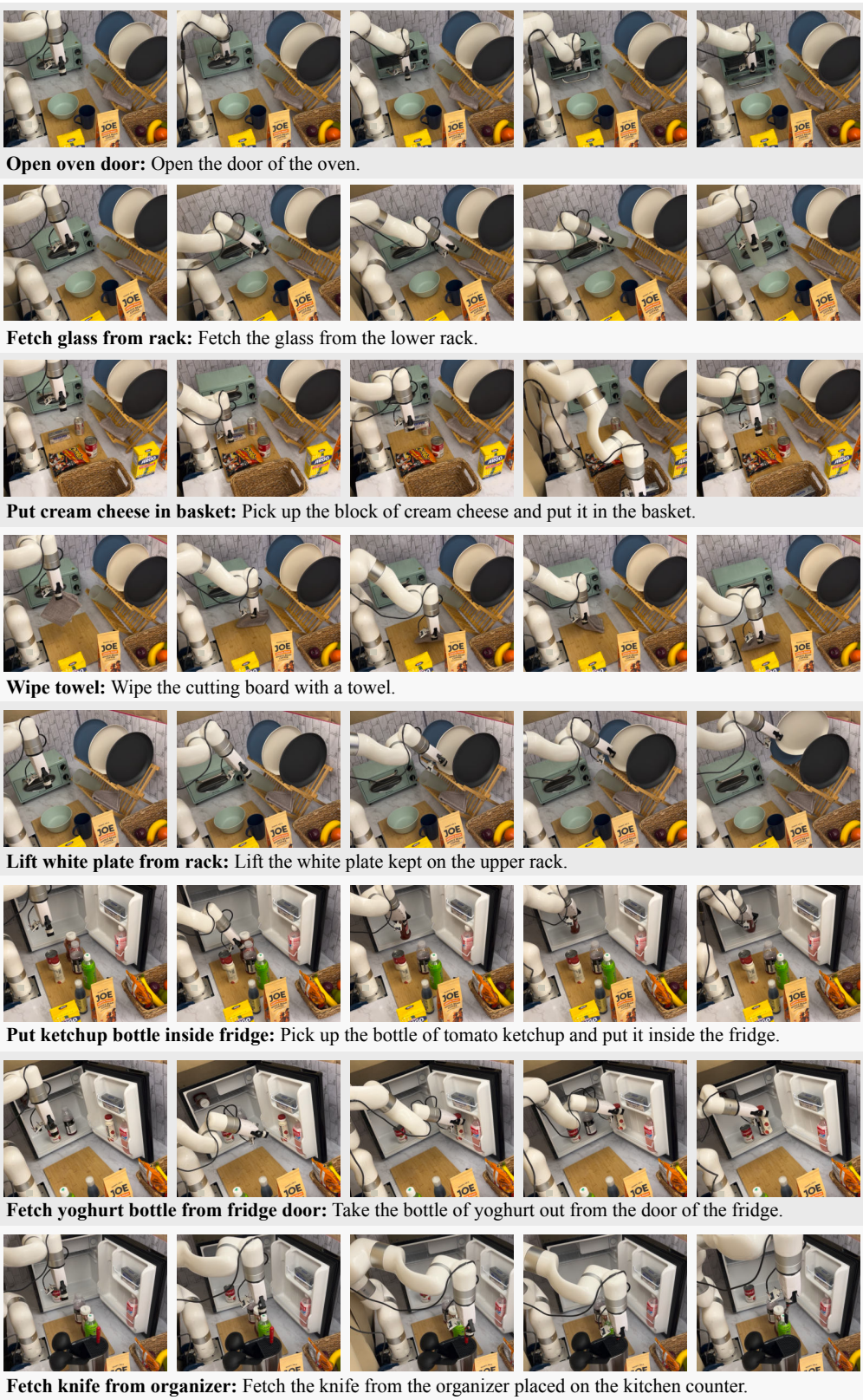


Figure 4: Real-world policy rollouts showing BAKU's capability in complex manipulation tasks.

https://doi.org/10.52202/079017-4484

Table 2: Performance of multi-task policies learned using BAKU on long-horizon tasks in the LIBERO-10 simulated benchmark and a real xArm robot. We observe that BAKU significantly outperforms prior work on both simulated and real world tasks.

Method	LIBERO-10 (10 tasks)	Real Robot (5 tasks)
MT-ACT	0.68	0.64
BAKU **(Ours)**	**0.86**	**0.84**

extended time horizons. Real world rollouts of these long-horizon tasks have been included in Appendix C with the task-wise performance and demonstration details in Appendix D.1.

4.4 What design decisions affect multi-task policy learning?

As described in Section 3, our multi-task policy architecture consists of three main components: sensory encoders, an observation trunk, and an action head. In this section, we analyze the design choices within each component and their effect on overall multi-task performance. We consider BAKU with an MLP action head (described in Section 3.4) as our base model. For ablations, we vary only a single property at a time while keeping all other aspects identical. This experimental setup allows us to clearly isolate the impact of individual design decisions. We examine different observation trunks, model sizes, action heads, goal modalities, and the use of action chunking, observation history, and task conditioning through FiLM [46]. The results of our ablation study are provided in Table 3 with more analysis in Appendix D. The results provide insights into which components and properties are most important for effective multi-task learning with BAKU.

Effect of Observation Trunk: We experiment with two trunk types: an MLP and a transformer architecture. In Table 3, we observe a slight performance dip when using an MLP trunk on Meta-World and DMC. For LIBERO-90, our most complex simulated benchmark, an MLP trunk resulted in a 9% lower success rate than a transformer trunk. This highlights the efficacy of transformers for modeling complex relationships between observations from multiple sensing modalities and actions.

Effect of Model Size: We study the effect of model size on multi-task performance by evaluating configurations with 4.4M, 10M, 31M, and 114M parameters. For each variant, we vary the size of the observation trunk and the action head while keeping the sensory encoders constant. The results in Table 3 show that the 4.4M, 10M, and 31M parameter models achieve similar performance across benchmarks. Surprisingly, the largest 114M parameter model severely underperforms on the harder LIBERO-90 benchmark. We suspect this poor performance may have been due to overfitting on the training data with a larger capacity. Based on these results, we use the 10M parameter model for BAKU since it is the smallest model with the best performance on 2 of the 3 simulated benchmarks.

Effect of Action Head: We compare the performance of BAKU retrofitted with five different action heads: MLP, GMM [36], BeT [60], VQ-BeT [31], and diffusion [10]. Having an independent action head enables us to extend these state-of-the-art action generation models to multi-task settings. Table 3 shows that on simulated benchmarks, a simple MLP head performs just as well or better than other multimodal action heads. Among the multimodal variants, VQ-BeT achieves the best performance. As a result, we also evaluate BAKU with a VQ-BeT action head in our real-world setup (Table 1). On the real robot, having a multimodal action head proves advantageous with the VQ-BeT head achieving a success rate of 91%, a 5% improvement over an MLP action head. Hence, our experiments demonstrate that while multimodal heads may provide benefits for real-world deployment, a simple MLP head can perform well on simulated data with limited behavioral diversity.

Effect of Action Chunking: We study the effect of action chunking on multi-task policy performance. For the image-based LIBERO-90 and Meta-World benchmarks, we predict a chunk of 10 future actions. For the locomotion tasks in DMC, we predicted 3 future actions. Based on the results in Table 3, we observe the largest difference on LIBERO-90, where removing action chunking and instead predicting a single action led to a 14% drop in performance. In contrast, there is no perceptible difference in performance on Meta-World with and without chunking. For the locomotion domains in

Table 3: Study of design decisions for BAKU that affects multi-task performance.

Category	Variant	LIBERO-90	Meta-World	DMC
Observation Trunk	MLP	0.81	0.78	0.68
	Transformer	**0.90**	**0.79**	**0.70**
Model Size	4.4M	0.85	0.78	0.68
	10M	**0.9**	0.79	**0.70**
	31M	0.87	**0.81**	**0.70**
	114M	0.19	**0.81**	0.68
Action Head	MLP	**0.90**	**0.79**	**0.70**
	GMM	0.84	0.65	0.67
	BeT	0.89	0.78	0.60
	VQ-BeT	**0.90**	0.78	**0.70**
	Diffusion	0.89	0.45	0.61
Action Chunking	✗	0.76	0.78	**0.74**
	✓	**0.90**	**0.79**	0.70
Observation History	✗	**0.90**	0.79	**0.70**
	With last-step loss	0.54	0.08	0.37
	With multi-step loss	**0.90**	**0.82**	0.68
Goal Modality	Text	0.90	0.79	N/A
	Goal Image	0.88	**0.81**	N/A
	Intermediate Image	**0.91**	0.80	N/A
FiLM	✗	0.87	**0.79**	N/A
	✓	**0.90**	**0.79**	N/A

DMC, we see a 4% performance increase when removing chunking. Hence, action chunking benefits manipulation tasks while mildly hindering locomotion tasks from our experiments.

Effect of Observation History: We study the effect of using an observation history on multi-task performance. As shown in Table 3, naively using an observation history where the action prediction loss is only computed for the last time step significantly degrades performance. However, since BAKU uses a transformer observation encoder, it allows predicting actions for all observations in the history and computing the prediction loss over all time steps. Empirically, we found this multi-step prediction loss provides richer supervision and improves the single-step loss performance by an average of 47% across all benchmarks. However, incorporating an observation history with multi-step action prediction did not noticeably improve overall policy performance compared to using no history. Therefore, our final architecture only uses the most recent observation as an input.

Effect of Goal Modality: We experiment with 3 different goal modalities: text instruction, goal image, and intermediate goal image. The text instructions are directly obtained from the task data. The goal image is obtained by randomly sampling a demonstration from the training dataset and taking the last frame. For an intermediate goal image, we consider this randomly sampled task demonstration, and for every time step, treat the frame k steps in the future as the goal image [72]. Table 3 contains the results on LIBERO-90 and Meta-World, as goal images do not apply to the state-based DMC tasks. We set k to 50 steps for LIBERO-90 and 30 steps for Meta-World. Since LIBERO-90 has two camera views, we use the third-person view to obtain goal images. We observe that all three goal modalities show a similar performance with slight variations. Overall, our approach supports different goal representations with only minor variations in performance.

Effect of FiLM Conditioning: We examine the impact of using a FiLM-conditioned vision encoder for language-guided multi-task policies. As shown in Table 3, on the image-based LIBERO-90 and Meta-World benchmarks, a FiLM-conditioned vision encoder performs equally well or better than an unconditional encoder. FiLM conditioning allows modulating the vision encoder's parameters conditioned on the language input. This provides an effective way to fuse visual and linguistic information for solving tasks. Therefore, BAKU employs a FiLM-conditioned vision encoder for our image-based experiments.

https://doi.org/10.52202/079017-4484

5 Related Work

Imitation Learning (IL) IL [23] refers to the setting where agents learn from demonstrations without access to environment rewards. IL can be broadly categorized into Behavior Cloning (BC) [48, 68] and Inverse Reinforcement Learning (IRL) [41, 1]. BC solely learns from offline demonstrations but suffers on out-of-distributions samples [56] whereas IRL focuses on learning a robust reward function through online interactions but suffers from sample inefficiency [17, 18]. In this work, we focus on using BC for learning multi-task policies. In recent years, there have been significant advances in single-task behavior cloning with the development of multimodal action generation models using GMMs [36, 39], EBMs [13], BeT [60, 11, 31], and diffusion [45, 10, 55, 7]. There has also been notable progress in solving long-horizon tasks through imitation learning with some works relying solely on robot data [38, 21, 9, 77, 33, 65] while others attempt to bootstrap learning from human data [72]. Further, these advances in policy learning combined with significant strides in self-supervised representation learning [8, 42, 20] have enabled deploying these policies in messy and unpredictable environments such as our homes [61] as well as zero-shot deployment in-the-wild [62, 64]. However, despite the large body of work advancing single-task robotic policy learning, there still exists a gap between single-task and multi-task performance for policy learning [75, 44, 57].

Multi-task Learning Robotics has a long history of multi-task learning. There is a significant body of work focusing on learning policies for robotic grasping with the aim of generalizing to new tasks [32, 47, 14, 71, 12], robotic language understanding [40, 35, 66, 3], and framing multi-task learning as a goal reaching problem [51, 28, 22]. Additionally, several works have collected varied multi-task robotics datasets [38, 34, 5, 43, 30]. Recently, there has been an increased use of transformer-based architectures for multi-task robot learning, spanning across robot navigation [62, 64], locomotion [27, 15, 49, 50], and manipulation [6, 78, 11, 5]. While most of these works use text conditioning for task specification, some go beyond text to use goal images [11, 16] and videos [26, 25] as well. Another emerging trend is co-training these robot policies with additional tasks, such as visual question answering and image captioning [52, 78], to develop more generalizable policies. Overall, multi-task learning has been widely applied in robotics and, more recently, using high-capacity transformer models to learn robot control policies has become common practice in the field. Despite their effectiveness, the architectures for these policies often become complicated, with the necessary components sometimes being unclear. Our proposed model, BAKU, combines key ideas from prior work into a single architecture to produce a model that is both simple and outperforms state-of-the-art methods in multi-task policy learning.

6 Conclusion and Limitations

In this work, we presented BAKU, a simple transformer architecture that demonstrates improved multi-task policy learning performance on a variety of simulated and real-world domains compared to prior state-of-the-art methods. We recognize a few limitations in this work: (a) In our real-world experiments, while BAKU achieved good performance on most tasks, it struggled on some precise manipulation tasks, such as *opening an oven door* or *placing a tea bottle in the fridge*. This suggests that data sharing across tasks of varying difficulty may hinder performance on more precise skills. Developing techniques to learn a single policy for different task complexity levels could help address this. (b) Currently, we focus on performing a single skill at a time. Developing algorithms capable of chaining multiple such skills can enable effective long-horizon robot manipulation. (c) In this work, we primarily studied the policy architecture and did not analyze the generalization benefits of multi-task learning as the number of tasks increases. A study of the emergence of such generalization with greater task diversity would be another interesting direction. Overall, we hope that BAKU serves as an important step towards developing multi-task policies capable of performing precise robotic manipulation.

Acknowledgments and Disclosure of Funding

We thank Nur Muhammad Shafiullah, Ulyana Piterbarg, Ademi Adeniji, Ben Evans, Gaoyue Zhou, and Irmak Güzey for valuable feedback and discussions. This work was supported by grants from Honda, Google, NSF award 2339096 and ONR awards N00014-21-1-2758 and N00014-22-1-2773. LP is supported by the Packard Fellowship.

References

[1] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1, 2004. 10

[2] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 1

[3] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022. 10

[4] J. Betker, G. Goh, L. Jing, T. Brooks, J. Wang, L. Li, L. Ouyang, J. Zhuang, J. Lee, Y. Guo, et al. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3):8, 2023. 1

[5] H. Bharadhwaj, J. Vakil, M. Sharma, A. Gupta, S. Tulsiani, and V. Kumar. Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking. *arXiv preprint arXiv:2309.01918*, 2023. 1, 3, 4, 5, 6, 10, 16, 17

[6] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022. 1, 3, 6, 10, 17

[7] L. Chen, S. Bahl, and D. Pathak. Playfusion: Skill acquisition via diffusion from language-annotated play. In *Conference on Robot Learning*, pages 2012–2029. PMLR, 2023. 10

[8] X. Chen, S. Xie, and K. He. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9640–9649, 2021. 10

[9] Y. Chen, C. Wang, L. Fei-Fei, and C. K. Liu. Sequential dexterity: Chaining dexterous policies for long-horizon manipulation. *arXiv preprint arXiv:2309.00987*, 2023. 10

[10] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023. 2, 3, 4, 8, 10, 16

[11] Z. J. Cui, Y. Wang, N. M. M. Shafiullah, and L. Pinto. From play to policy: Conditional behavior generation from uncurated robot data. *arXiv preprint arXiv:2210.10047*, 2022. 3, 10

[12] H.-S. Fang, C. Wang, H. Fang, M. Gou, J. Liu, H. Yan, W. Liu, Y. Xie, and C. Lu. Anygrasp: Robust and efficient grasp perception in spatial and temporal domains. *IEEE Transactions on Robotics*, 2023. 10

[13] P. Florence, C. Lynch, A. Zeng, O. A. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson. Implicit behavioral cloning. In *Conference on Robot Learning*, pages 158–168. PMLR, 2022. 10

[14] A. Gupta, A. Murali, D. P. Gandhi, and L. Pinto. Robot learning in homes: Improving generalization and reducing dataset bias. *Advances in neural information processing systems*, 31, 2018. 10

[15] A. Gupta, L. Fan, S. Ganguli, and L. Fei-Fei. Metamorph: Learning universal controllers with transformers. *arXiv preprint arXiv:2203.11931*, 2022. 10

[16] S. Haldar and L. Pinto. Polytask: Learning unified policies through behavior distillation. *arXiv preprint arXiv:2310.08573*, 2023. 10

[17] S. Haldar, V. Mathur, D. Yarats, and L. Pinto. Watch and match: Supercharging imitation with regularized optimal transport. In *Conference on Robot Learning*, pages 32–43. PMLR, 2023. 5, 10

 https://doi.org/10.52202/079017-4484

[18] S. Haldar, J. Pari, A. Rai, and L. Pinto. Teach a robot to fish: Versatile imitation from one minute of demonstrations. *arXiv preprint arXiv:2303.01497*, 2023. 5, 10

[19] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[20] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022. 10

[21] M. Heo, Y. Lee, D. Lee, and J. J. Lim. Furniturebench: Reproducible real-world benchmark for long-horizon complex manipulation. *arXiv preprint arXiv:2305.12821*, 2023. 10

[22] D.-A. Huang, Y.-W. Chao, C. Paxton, X. Deng, L. Fei-Fei, J. C. Niebles, A. Garg, and D. Fox. Motion reasoning for goal-based imitation learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4878–4884. IEEE, 2020. 10

[23] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(2):1–35, 2017. 10

[24] A. Iyer, Z. Peng, Y. Dai, I. Guzey, S. Haldar, S. Chintala, and L. Pinto. Open teach: A versatile teleoperation system for robotic manipulation. *arXiv preprint arXiv:2403.07870*, 2024. 3, 5

[25] V. Jain, M. Attarian, N. J. Joshi, A. Wahid, D. Driess, Q. Vuong, P. R. Sanketi, P. Sermanet, S. Welker, C. Chan, I. Gilitschenski, Y. Bisk, and D. Dwibedi. Vid2robot: End-to-end video-conditioned policy learning with cross-attention transformers, 2024. 10

[26] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2022. 10

[27] M. Janner, Q. Li, and S. Levine. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems*, 34:1273–1286, 2021. 10

[28] T. Jurgenson, O. Avner, E. Groshev, and A. Tamar. Sub-goal trees a framework for goal-based reinforcement learning. In *International conference on machine learning*, pages 5020–5030. PMLR, 2020. 10

[29] A. Karpathy. mingpt: A minimal pytorch re-implementation of the openai gpt. https://github.com/karpathy/minGPT, 2021. 4, 18

[30] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024. 10

[31] S. Lee, Y. Wang, H. Etukuru, H. J. Kim, N. M. M. Shafiullah, and L. Pinto. Behavior generation with latent actions. *arXiv preprint arXiv:2403.03181*, 2024. 2, 3, 4, 6, 8, 10, 16

[32] I. Lenz, H. Lee, and A. Saxena. Deep learning for detecting robotic grasps. *The International Journal of Robotics Research*, 34(4-5):705–724, 2015. 10

[33] T. Lin, Y. Zhang, Q. Li, H. Qi, B. Yi, S. Levine, and J. Malik. Learning visuotactile skills with two multifingered hands. *arXiv:2404.16823*, 2024. 10

[34] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36, 2024. 2, 4, 5, 10, 17

[35] C. Lynch and P. Sermanet. Language conditioned imitation learning over unstructured data. *arXiv preprint arXiv:2005.07648*, 2020. 10

[36] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet. Learning latent plans from play. In *Conference on robot learning*, pages 1113–1132. PMLR, 2020. 4, 8, 10, 16

[37] A. Mandlekar, Y. Zhu, A. Garg, J. Booher, M. Spero, A. Tung, J. Gao, J. Emmons, A. Gupta, E. Orbay, et al. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. In *Conference on Robot Learning*, pages 879–893. PMLR, 2018. 1

[38] A. Mandlekar, D. Xu, R. Martín-Martín, S. Savarese, and L. Fei-Fei. Learning to generalize across long-horizon tasks from human demonstrations. *arXiv preprint arXiv:2003.06085*, 2020. 10

[39] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021. 2, 10

[40] H. Mei, M. Bansal, and M. Walter. Listen, attend, and walk: Neural mapping of navigational instructions to action sequences. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016. 10

[41] A. Y. Ng, S. J. Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000. 10

[42] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 10

[43] A. Padalkar, A. Pooley, A. Jain, A. Bewley, A. Herzog, A. Irpan, A. Khazatsky, A. Rai, A. Singh, A. Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023. 10

[44] E. Parisotto, J. L. Ba, and R. Salakhutdinov. Actor-mimic: Deep multitask and transfer reinforcement learning. *arXiv preprint arXiv:1511.06342*, 2015. 1, 10

[45] T. Pearce, T. Rashid, A. Kanervisto, D. Bignell, M. Sun, R. Georgescu, S. V. Macua, S. Z. Tan, I. Momennejad, K. Hofmann, et al. Imitating human behaviour with diffusion models. *arXiv preprint arXiv:2301.10677*, 2023. 4, 10, 16

[46] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018. 1, 4, 8, 16

[47] L. Pinto and A. Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 3406–3413. IEEE, 2016. 10

[48] D. Pomerleau. An autonomous land vehicle in a neural network. *Advances in Neural Information Processing Systems*, 1, 1998. 10

[49] I. Radosavovic, T. Xiao, B. Zhang, T. Darrell, J. Malik, and K. Sreenath. Learning humanoid locomotion with transformers. *arXiv e-prints*, pages arXiv–2303, 2023. 10

[50] I. Radosavovic, B. Zhang, B. Shi, J. Rajasegaran, S. Kamat, T. Darrell, K. Sreenath, and J. Malik. Humanoid locomotion as next token prediction. *arXiv preprint arXiv:2402.19469*, 2024. 10

[51] A. Raffin, A. Hill, R. Traoré, T. Lesort, N. Díaz-Rodríguez, and D. Filliat. Decoupling feature extraction from policy learning: assessing benefits of state representation learning in goal based robotics. *arXiv preprint arXiv:1901.08651*, 2019. 10

[52] S. Reed, K. Zolna, E. Parisotto, S. G. Colmenarejo, A. Novikov, G. Barth-Maron, M. Gimenez, Y. Sulsky, J. Kay, J. T. Springenberg, et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022. 10

[53] M. Reid, N. Savinov, D. Teplyashin, D. Lepikhin, T. Lillicrap, J.-b. Alayrac, R. Soricut, A. Lazaridou, O. Firat, J. Schrittwieser, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024. 1

 https://doi.org/10.52202/079017-4484

[54] N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019. URL https://arxiv.org/abs/1908.10084. 4

[55] M. Reuss, M. Li, X. Jia, and R. Lioutikov. Goal-conditioned imitation learning using score-based diffusion policies. *arXiv preprint arXiv:2304.02532*, 2023. 4, 10, 16

[56] S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011. 10

[57] A. A. Rusu, S. G. Colmenarejo, C. Gulcehre, G. Desjardins, J. Kirkpatrick, R. Pascanu, V. Mnih, K. Kavukcuoglu, and R. Hadsell. Policy distillation. *arXiv preprint arXiv:1511.06295*, 2015. 1, 10

[58] M. Ryoo, A. Piergiovanni, A. Arnab, M. Dehghani, and A. Angelova. Tokenlearner: Adaptive space-time tokenization for videos. *Advances in neural information processing systems*, 34: 12786–12797, 2021. 6

[59] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, K. Ghasemipour, R. Gontijo Lopes, B. Karagol Ayan, T. Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35: 36479–36494, 2022. 1

[60] N. M. Shafiullah, Z. Cui, A. A. Altanzaya, and L. Pinto. Behavior transformers: Cloning k modes with one stone. *Advances in neural information processing systems*, 35:22955–22968, 2022. 2, 4, 8, 10, 16

[61] N. M. M. Shafiullah, A. Rai, H. Etukuru, Y. Liu, I. Misra, S. Chintala, and L. Pinto. On bringing robots home. *arXiv preprint arXiv:2311.16098*, 2023. 1, 10

[62] D. Shah, A. Sridhar, N. Dashora, K. Stachowicz, K. Black, N. Hirose, and S. Levine. Vint: A foundation model for visual navigation. *arXiv preprint arXiv:2306.14846*, 2023. 10

[63] M. Shridhar, L. Manuelli, and D. Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023. 3

[64] A. Sridhar, D. Shah, C. Glossop, and S. Levine. Nomad: Goal masked diffusion policies for navigation and exploration. *arXiv preprint arXiv:2310.07896*, 2023. 10

[65] K. Sridhar, S. Dutta, D. Jayaraman, J. Weimer, and I. Lee. Memory-consistent neural networks for imitation learning. *arXiv preprint arXiv:2310.06171*, 2023. 10

[66] S. Stepputtis, J. Campbell, M. Phielipp, S. Lee, C. Baral, and H. Ben Amor. Language-conditioned imitation learning for robot manipulation tasks. *Advances in Neural Information Processing Systems*, 33:13139–13150, 2020. 10

[67] Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. d. L. Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018. 2, 5, 17

[68] F. Torabi, G. Warnell, and P. Stone. Recent advances in imitation learning from observation. *arXiv preprint arXiv:1905.13566*, 2019. 2, 10

[69] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 1

[70] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 4

[71] U. Viereck, A. Pas, K. Saenko, and R. Platt. Learning a visuomotor controller for real world robotic grasping using simulated depth images. In *Conference on robot learning*, pages 291–300. PMLR, 2017. 10

[72] C. Wang, L. Fan, J. Sun, R. Zhang, L. Fei-Fei, D. Xu, Y. Zhu, and A. Anandkumar. Mimicplay: Long-horizon imitation learning by watching human play. In *7th Annual Conference on Robot Learning*, 2023. 3, 9, 10

[73] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers, 2020. 4

[74] D. Yarats, R. Fergus, A. Lazaric, and L. Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*, 2021. 5

[75] T. Yu, S. Kumar, A. Gupta, S. Levine, K. Hausman, and C. Finn. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33:5824–5836, 2020. 1, 10

[76] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pages 1094–1100. PMLR, 2020. 2, 5, 17

[77] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023. 2, 3, 4, 5, 10, 16, 17

[78] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023. 10

 https://doi.org/10.52202/079017-4484

A Algorithmic Details

A.1 FiLM Conditioning

Feature-wise Linear Modulation (FiLM) [46] is a technique used for conditioning neural networks that allows the network to modulate its behavior based on an external conditioning signal, such as text instructions or observations. In the context of text conditioning for policy learning, the text instructions are first encoded into a conditioning vector. This conditioning vector is then used to modulate the activations of the neural network through FiLM layers. FiLM applies a feature-wise affine transformation (scaling and shifting) to the activations of the network, conditioned on the text embedding. In other word, assuming $\mathbf{x}$ is a FiLM layer's input, $\mathbf{z}$ is a conditioning input, and γ and β are $\mathbf{z}$-dependent scaling and shifting vectors,

$$FiLM(\mathbf{x}) = \gamma(\mathbf{z}) \odot \mathbf{x} + \beta(\mathbf{z}) \tag{3}$$

This allows the network to adapt its computation and output based on the given text instructions, enabling tasks like instruction following or conditioning the policy on language descriptions.

A.2 Action Heads

Having a separate action prediction module allows BAKU to leverage state-of-the-art techniques for action generation. In this work, we evaluate five different action head variants. Below we briefly describe each variant. For more details on these methods, please refer to the original publications.

Multilayer Perceptron (MLP) This is a simple neural network comprising multiple dense layers. We use a two-layer MLP for our experiments.

Gaussian Mixture Model (GMM) [36] A Gaussian mixture model (GMM) action head models the policy as a mixture of Gaussians, enabling multi-modal action sampling for continuous control problems. The GMM parameters are part of the learned policy network. For our experiments, we employ a two-layer GMM action head with five action modes and a Softplus activation function.

Behavior Transformer (BeT) [60] The Behavior Transformer (BeT) models continuous action prediction as a two-part problem. Actions in the training data are first clustered into k bins using k-means clustering. A discrete action head classifies the cluster an action belongs to, while an offset action head predicts an offset value added to the corresponding cluster center. The discrete head uses a focal loss, while the offset head uses L2 loss. For our experiments, we use BeT with 64 action clusters.

Vector-Quantized Behavior Transformer (VQ-BeT) [31] The Vector-Quantized Behavior Transformer (VQ-BeT) extends BeT by replacing k-means clustering with residual VQVAE-based tokenization, significantly improving performance over BeT. For our experiments, we employ VQ-BeT with two residual VQ layers of codebook size and latent dimension 16 and 256, respectively.

Diffusion [45, 10, 55] A diffusion action head models action prediction as a diffusion process that generates actions over time by iteratively denoising samples from a Gaussian distribution. While highly effective for multi-modal distributions, the iterative denoising during inference slows deployment speed. In this work, we use a transformer-based diffusion head introduced by prior work [45, 10]. We use a two-layer diffusion head for our experiments.

A.3 Temporal smoothing over action chunking

A naïve implementation of action chunking, where a new environment observation in incorporated every k steps can be suboptimal and can result in jerky robot motion. To improve the smoothness in robot motion, we incorporate an exponential temporal ensembling technique, following prior work [77, 5]. Instead of querying the policy every k steps, we query it at every timestep. This results in an overlap in predicted action chunks and at any given timestep, there will be more than one predicted actions. Instead of using only the current action prediction, we use a temporal ensemble to

combine all the past predictions. This temporal ensemble performs a weighted average over these predictions with an exponential weighing scheme $w_i = exp(-m * i)$, where w_0 is the weight for the oldest action. The speed for incorporating a new observation is governed by m, where a smaller m means faster incorporation. It must be noted that this ensembling incurs no additional training cost, only extra inference-time computation. In our experiments, similar to prior work [77, 5], we find both action chunking and temporal ensembling to be important for producing precise and smooth motion.

A.4 Hyperparameters

The complete list of hyperparameters is provided in Table 4. For RT-1 [6], we use our implementation with an RT-1 action head that discretizes the continuous action into discrete bins uniformly. For MT-ACT [5], we use the open-source implementation with the default hyperparameters. We vary the action chunk length for MT-ACT for different benchmarks, the values for which have been provided in Table 4.

Training time Below we provide details about the time required to train BAKU on a single NVIDIA RTX A4000 GPU.

1. **LIBERO:** Training for $600k$ steps with a batch size of 64 and 2 camera views and robot proprioception as input requires around 10.5 hours.
2. **Meta-World:** Training for $600k$ steps with a batch size of 64 and 1 camera view as input requires around 8 hours.
3. **DM Control:** Training for $2M$ steps with a batch size of 128 and robot state as input requires around 26 hours.
4. **xArm Robot:** Training for $200k$ steps with a batch size of 64 and 4 camera views and robot proprioception as input requires around 6 hours.

B Simulation Tasks

We evaluate BAKU on three simulated benchmarks: LIBERO-90 [34], MetaWorld [76], and DM Control [67]. For LIBERO-90, we directly use the dataset provided, which includes demonstrations for all 90 tasks. For details on the specific LIBERO-90 tasks, please refer to the original paper [34]. For MetaWorld and DM Control, we collected demonstrations from expert agents trained with reinforcement learning (RL). We include only the tasks for which we were able to obtain expert demonstration data. Table 5 lists the 30 MetaWorld tasks and 9 DM Control tasks used in our experiments.

C Robot Tasks

We evaluate BAKU on 30 tasks in our real-world multi-task kitchen environment. We provide the task description along with policy deployment rollouts with BAKU for each task in Figures 5, 6, 7, 8, and 9. The long-horizon task rollouts have been shown in Figure 10.

Robot control We deploy our learned policies at 10Hz using a high-level controller. To facilitate smooth motion on the robot, we deploy a low-level Minimum-Jerk Controller at 100Hz.

D Additional Results and Analysis

D.1 Real-World Task-wise Results

Table 6 provides the task-wise performance for all 30 tasks in our real-world multi-task kitchen environment. We collect an average of 17 demonstrations per task, with a total of 520 demonstrations across all tasks. Task-wise performance for the real-world long-horizon tasks has been included in Table 7.

https://doi.org/10.52202/079017-4484

Table 4: List of hyperparameters.

Method	Parameter	Value
Common	Learning rate	$1e^{-4}$
	Image size	128×128 (LIBERO-90, xArm)
		84×84 (Meta-World)
	Mini-batch size	64 (LIBERO-90, Meta-World, xArm)
		128 (DM Control)
	Optimizer	Adam
	Number of training steps	600000 (LIBERO-90, Meta-World)
		2000000 (DM Control)
		200000 (xArm)
	Number of demonstrations	50 (LIBERO-90)
		35 (Meta-World)
		500 (DM Control)
		15 (xArm)
	Transformer architecture	minGPT [29] (with 8 layers and 4 heads)
	Action chunk length	10 (LIBERO-90, Meta-World)
		3 (DMC)
		20 (xArm)
BAKU	Observation trunk	Transformer
	Action head	MLP (base)
		GMM, BeT, VQ-BeT, Diffusion (variants)
	Hidden dim	256
	Observation history	False
	Action chunking	True
	Intermediate goal steps (k)	50 (LIBERO-90)
		30 (Meta-World)
RT-1	Observation trunk	Transformer
	Action head	MLP (base)
	Hidden dim	512
	Observation history	True
	History length	6
	Action chunking	False
MT-ACT	Observation history	False
	Action chunking	True

Table 5: List of tasks in Meta-World and DM Control.

Meta-World	DM Control
basketball-v2	cartpole swingup
bin-picking-v2	cheetah run
button-press-v2	hopper stand
button-press-topdown-v2	quadruped run
button-press-topdown-wall-v2	quadruped walk
button-press-wall-v2	teacher easy
coffee-button-v2	walker stand
coffee-pull-v2	walker walk
coffee-push-v2	walker run
dial-turn-v2	
disassemble-v2	
door-lock-v2	
door-open-v2	
door-unlock-v2	
drawer-close-v2	
drawer-open-v2	
faucet-close-v2	
faucet-open-v2	
hammer-v2	
handle-press-v2	
handle-press-side-v2	
handle-pull-v2	
handle-pull-side-v2	
peg-insert-side-v2	
peg-unplug-side-v2	
plate-slide-v2	
plate-slide-back-v2	
plate-slide-back-side-v2	
plate-slide-side-v2	
shelf-place-v2	
soccer-v2	
stick-push-v2	
sweep-v2	
sweep-into-v2	
window-close-v2	
window-open-v2	

D.2 Additional Analysis

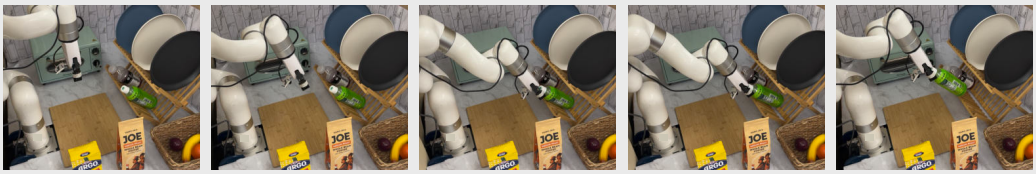
In addition to the analysis in Section 4.4, we provide further comparisons here to better justify our design choices.

Separate vs. Shared Vision Encoders On the LIBERO-90 benchmark, environment observations include images from two camera views. Table 12 compares multi-task performance using either a common encoder for both views or separate view-specific encoders. While separate encoders provide a 2% boost in performance, this minor gain comes at the cost of a 15% increase in parameter count per camera view added (since the visual encoders comprise 1.5M parameters in our 10M parameter model). For our real-world experiments involving 4 camera views, this parameter increase would be even more significant. Therefore, in BAKU, we use a shared encoder for all views to keep the model compact, assisting with faster inference speeds.

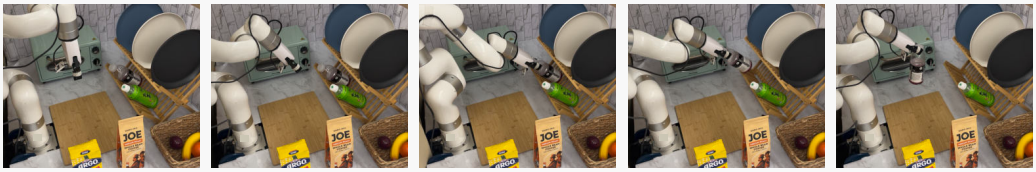
 https://doi.org/10.52202/079017-4484

Fetch glass from rack: Fetch the glass from the lower rack.

Fetch towel from rack: Fetch the towel from the lower rack.

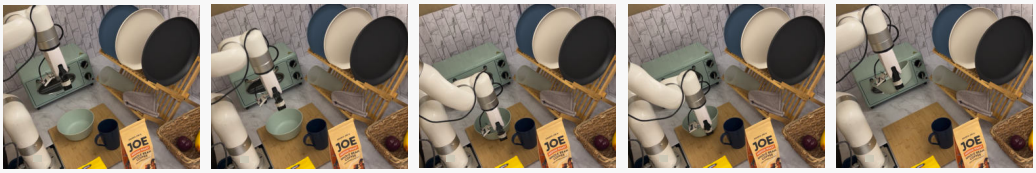


Fetch tea bottle from rack: Fetch the bottle of green tea from the lower rack.

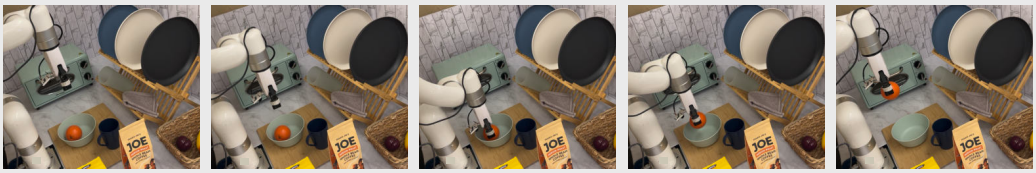


Fetch water bottle from rack: Fetch the bottle of vitamin water from the lower rack.

Pick blue mug: Pick up the blue mug from the kitchen counter.



Pick light blue bowl: Pick up the light blue bowl from the kitchen counter.



Pick orange from bowl: Pick up the orange from inside the light blue bowl kept on the kitchen counter.

Figure 5: Real-world policy rollouts showing BAKU's capability in complex manipulation tasks.

Data Efficiency Analysis We analyze the performance of BAKU with varying number of demonstrations in Table 8 and Table 9. We observe that at each level of data availability, BAKU shows a significantly higher success rate than MT-ACT and RT-1.

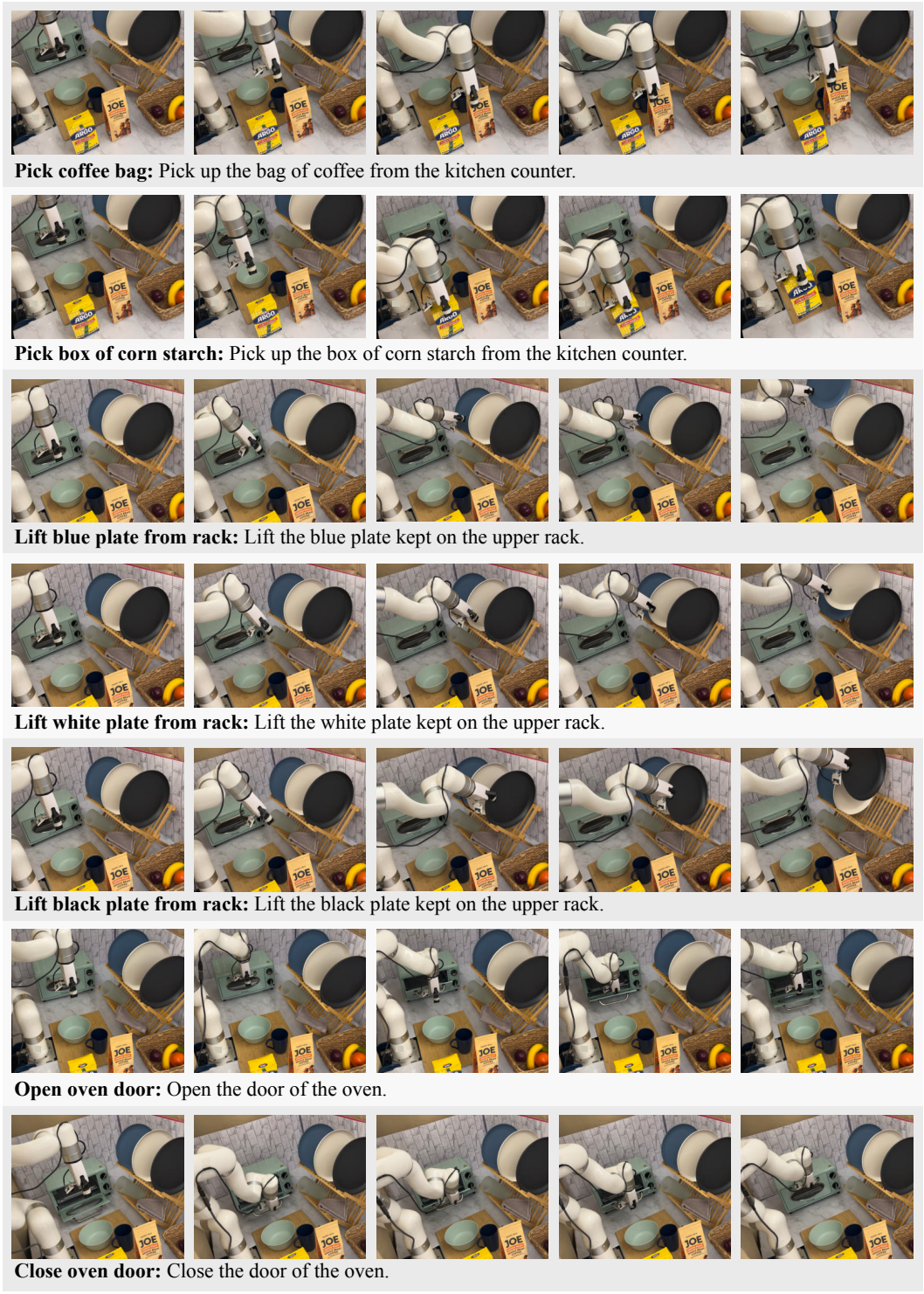


Figure 6: Real-world policy rollouts showing BAKU's capability in complex manipulation tasks.

Robustness to training seeds We provide results on BAKU, RT-1, and MT-ACT across 3 seeds in Table 10. We observe that all three methods are robust to different seed values. Further, probabilistic approaches like GMM and diffusion might be sensitive to favorable seed values, and evaluating on a single seed might make the result unreliable. Thus, Table 11 includes results across 3 seeds on BAKU

https://doi.org/10.52202/079017-4484

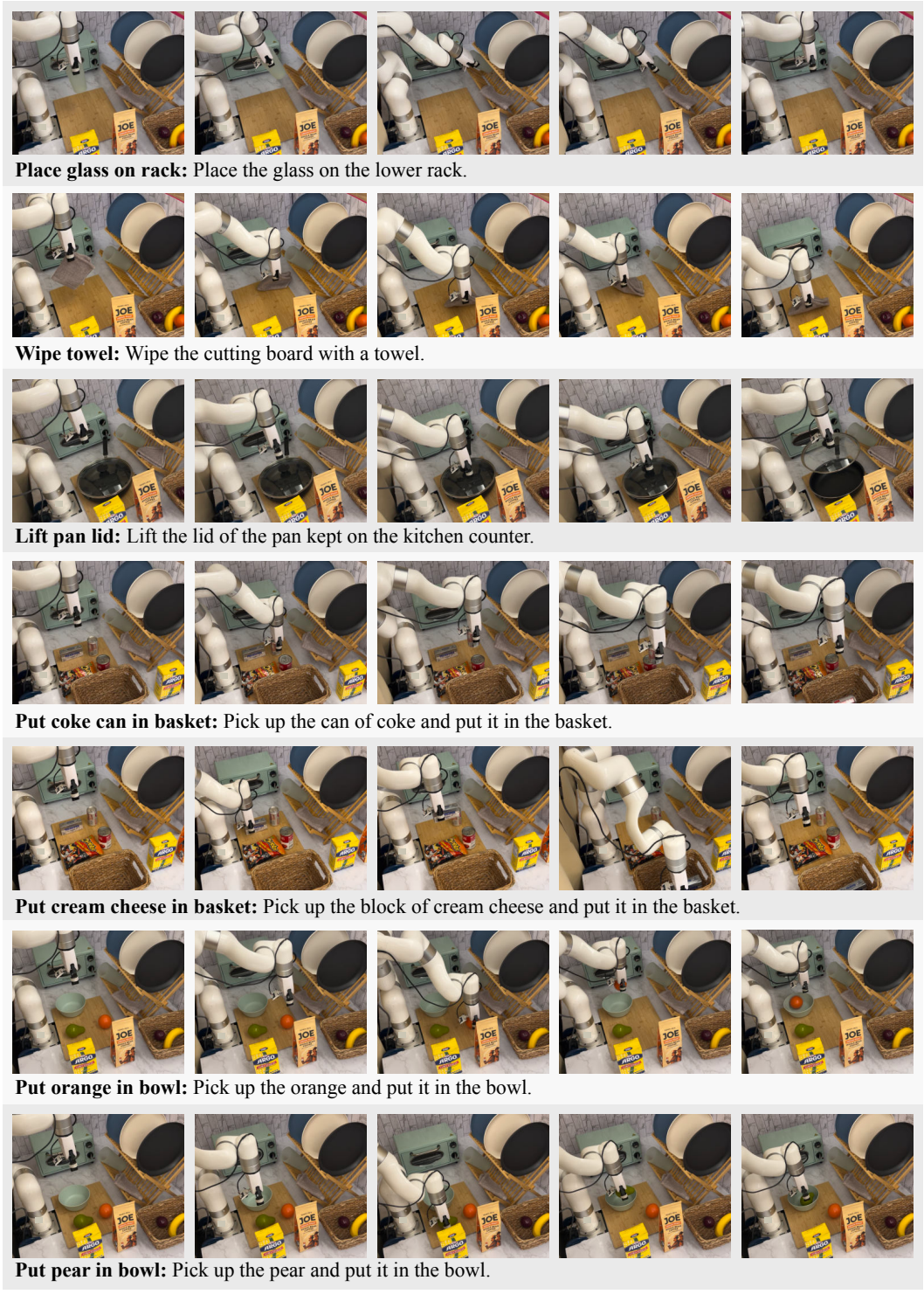


Figure 7: Real-world policy rollouts showing BAKU's capability in complex manipulation tasks.

with different multimodal heads. We observe that BAKU with different action heads is robust to the value of the random seed. Due to limited compute and the large number of multi-task experiments, we provide these results on the LIBERO-90 and Metaworld benchmarks.

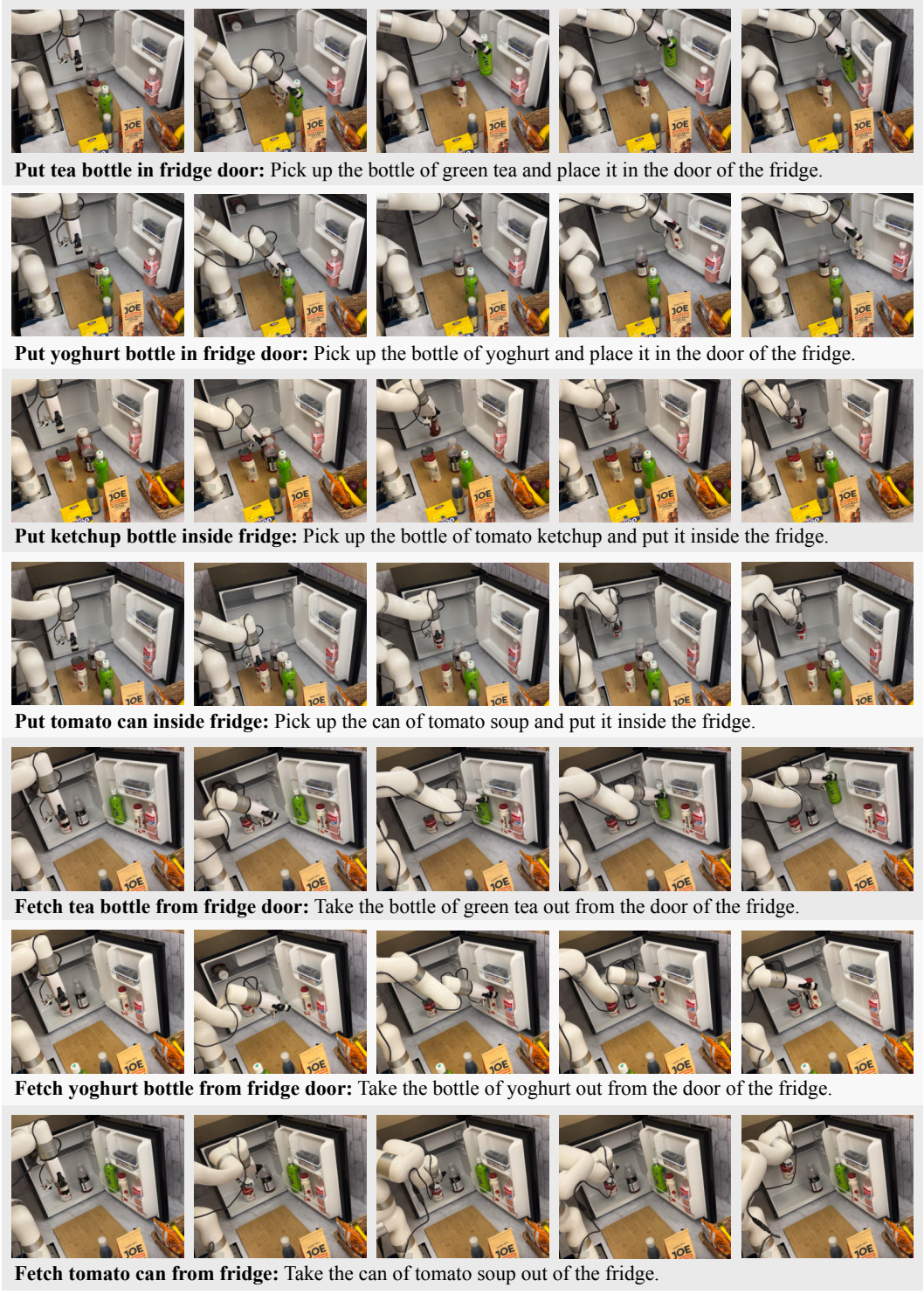
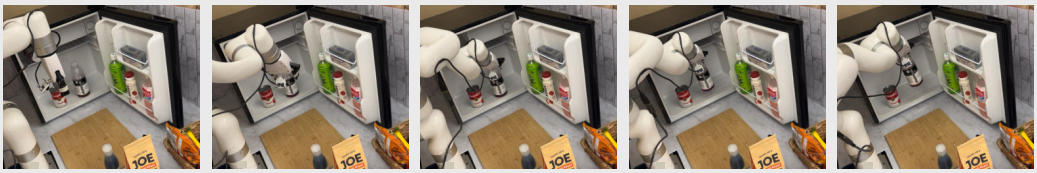


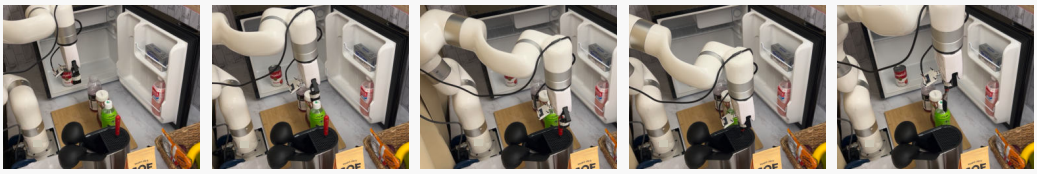
Figure 8: Real-world policy rollouts showing BAKU's capability in complex manipulation tasks.

Observation trunk input In our proposed architecture (see Section 3.4), the encoded observations from different modalities are passed individually as tokens into the observation trunk along with the action token to output the action feature representation. An alternative approach is to concatenate all the encoded inputs into a single vector and pass it through the observation trunk. As shown in Table 12, for Meta-World and DMC, which each have only a single input source, there is no

https://doi.org/10.52202/079017-4484

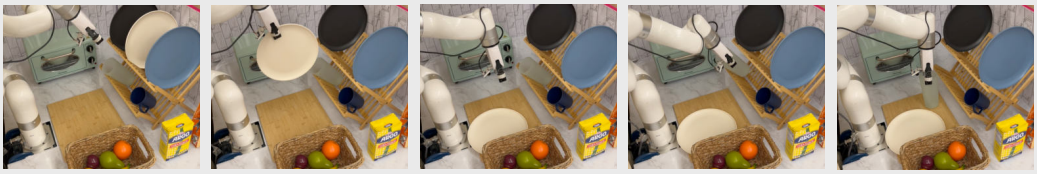


Fetch water bottle from fridge: Take the bottle of vitamin water out of the fridge.

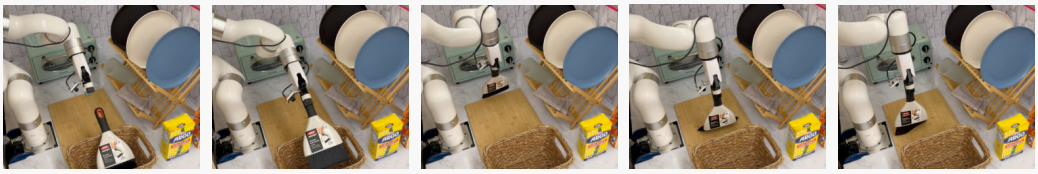


Fetch knife from organizer: Fetch the knife from the organizer placed on the kitchen counter.

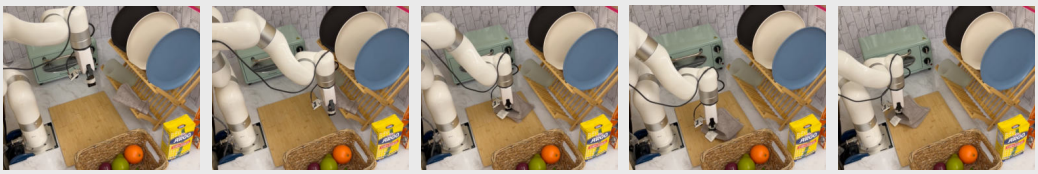
Figure 9: Real-world policy rollouts showing BAKU's capability in complex manipulation tasks.



Set up table: Place the white plate and the glass on the kitchen counter.

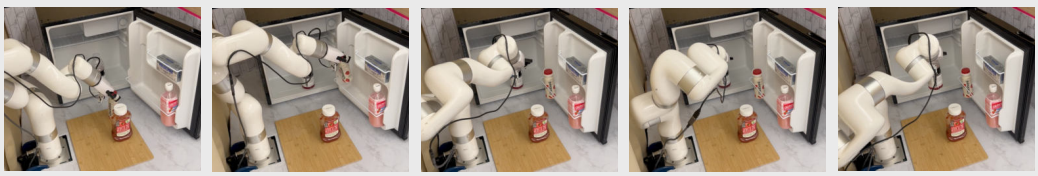


Pick broom and sweep: Pick up the broom and sweep the cutting board.



Pick towel and wipe: Pick up the towel from the lower rack and wipe the cutting board.

Take bowl out of the oven: Take the bowl out of the oven and place it on the kitchen counter.



Put yoghurt inside and take water bottle out of fridge: Put the yoghurt in the door of the fridge and take out the bottle of water from inside the fridge.

Figure 10: Real-world policy rollouts showing BAKU's capability on long-horizon manipulation tasks.

Table 6: Real task-wise performance

Task	Number of Demonstrations	Successes (out of 5)			
		RT-1	MTACT	Baku	Baku w/ VQ-BeT
Fetch glass from rack	20	5	5	5	5
Fetch towel from rack	28	5	2	5	5
Fetch tea bottle from rack	16	0	3	5	5
Fetch water bottle from rack	16	0	0	5	5
Pick blue mug	16	5	5	5	5
Pick light blue bowl	25	5	5	5	5
Pick orange from bowl	27	0	0	3	4
Pick coffee bag	19	3	5	5	5
Pick box of corn starch	14	0	3	5	5
Lift blue plate from the rack	18	0	4	5	5
Lift white plate from the rack	18	5	5	5	5
Lift black plate from the rack	12	2	3	5	5
Open oven door	17	0	0	0	3
Close oven door	27	0	3	3	4
Place glass on rack	19	5	5	5	5
Wipe towel	17	4	5	5	5
Lift pan lid	18	1	2	4	4
Put coke can in basket	19	0	0	3	3
Put cream cheese in basket	19	0	3	5	5
Put orange into bowl	14	0	0	4	5
Put pear into bowl	17	0	0	3	5
Put tea bottle in fridge door	18	0	0	1	0
Put yoghurt bottle in fridge door	17	3	5	3	5
Put ketchup bottle inside fridge	15	5	4	5	5
Put tomato can inside fridge	11	0	0	5	4
Fetch tea bottle from fridge door	11	5	5	5	5
Fetch tomato can from fridge door	11	0	1	5	5
Fetch yoghurt bottle from fridge door	10	0	3	5	4
Fetch water bottle from fridge	11	2	3	5	5
Fetch knife from organizer	20	0	5	5	5
Mean	17	1.83	2.8	4.3	**4.53**
Mean success rate (out of 1)	–	0.37	0.56	0.86	**0.91**

difference in performance, as expected. However, for LIBERO-90, which uses two camera views and the robot's proprioceptive state as inputs, there is a 3% absolute improvement in performance when using separate observation tokens as compare to a single concatenated vector.

E Broader Impacts

In this work, we present BAKU, a simple and efficient transformer architecture for multi-task policy learning. This work takes an important step toward enabling more efficient training of generalist robotic agents capable of performing diverse tasks, reducing the need for large datasets of expert demonstrations which are costly and time-consuming to collect. Further, BAKU focuses on improving data efficiency by maximally leveraging available training data, which is particularly valuable in robotics where data collection is expensive.

https://doi.org/10.52202/079017-4484

Table 7: Real task-wise performance for long-horizon tasks

Task	Number of Demonstrations	Successes (out of 5)	
		MTACT	Baku
Set up table	34	3	3
Pick broom and sweep	13	4	5
Pick towel and wipe	14	2	4
Take bowl out of the oven	18	5	5
Put yoghurt inside and take water bottle out of fridge	17	2	4
Mean	19	3.2	**4.2**
Mean success rate (out of 1)	–	0.64	**0.84**

Table 8: Data efficiency analysis on the LIBERO-90 benchmark.

# Demos	RT-1	MT-ACT	BAKU
5	0	0.31	**0.58**
10	0.01	0.48	**0.71**
25	0.04	0.49	**0.83**
50	0.16	0.54	**0.9**

Table 9: Data efficiency analysis on the Meta-World benchmark.

# Demos	RT-1	MT-ACT	BAKU
5	0.40	0.07	**0.59**
10	0.49	0.10	**0.67**
25	0.62	0.11	**0.76**
35	0.65	0.13	**0.79**

Table 10: Performance of multi-task policies learned using BAKU on LIBERO-90 and Meta-World. We report the mean and standard deviation for each variant across 3 seeds.

Method	LIBERO-90 (90 tasks)	Meta-World (30 tasks)
RT-1	0.14 ± 0.02	0.64 ± 0.01
MTACT	0.55 ± 0.01	0.12 ± 0.01
BAKU **(Ours)**	$\mathbf{0.89 \pm 0.01}$	$\mathbf{0.81 \pm 0.02}$

Table 11: Performance of BAKU with different action heads on LIBERO-90 and Meta-World. We report the mean and standard deviation for each variant across 3 seeds.

Action Head	LIBERO-90	Meta-World
MLP	0.89 ± 0.01	$\mathbf{0.81 \pm 0.02}$
GMM	0.83 ± 0.02	0.64 ± 0.02
BeT	0.88 ± 0.01	0.77 ± 0.01
VQ-BeT	$\mathbf{0.9 \pm 0.01}$	0.78 ± 0.005
Diffusion	0.88 ± 0.01	0.64 ± 0.01

Table 12: Study of design decisions for the model architecture that affects multi-task performance.

Category	Variant	LIBERO-90	Meta-World	DMC
Separate vs. Shared Vision Encoders	Common	0.90	–	–
	Separate	**0.92**	–	–
Observation Trunk Input	Separate	**0.90**	**0.79**	**0.70**
	Concatenated	0.87	**0.79**	**0.70**

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: Section 3 explain the architecture in detail with the performance and ablations presented in Section 4.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: The limitations have been discussed in Section 6.

   Guidelines:

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

3. **Theory Assumptions and Proofs**

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

   Answer: [NA]

https://doi.org/10.52202/079017-4484

Justification: The paper does not include theoretical results.

Guidelines:

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

4. **Experimental Result Reproducibility**

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

   Answer: [Yes]

   Justification: We include our architecture details in Section 3.4 and experimental details in Section 4 and Appendix A.4. We will also be making our code and data public on the project website.

   Guidelines:

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

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

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

Answer: [Yes]

Justification: We will also be making our code and data public on the project website.

Guidelines:

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

6. **Experimental Setting/Details**

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

   Answer: [Yes]

   Justification: We include our architecture details in Section 3.4 and all experimental details in Section 4 and Appendix A.4. We will also be making our code and data public on the project website.

   Guidelines:

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

7. **Experiment Statistical Significance**

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

   Answer: [Yes]

   Justification: Error bars are not reported because of insufficient computational resources.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

 https://doi.org/10.52202/079017-4484

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

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

   Answer: [Yes]

   Justification: The details have been included in Appendix A.4.

   Guidelines:

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

9. **Code Of Ethics**

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

   Answer: [Yes] .

   Justification: The paper conforms with the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader Impacts**

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

    Answer: [Yes]

    Justification: We include of broader impacts statement of our work in Appendix E.

    Guidelines:

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

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

11. **Safeguards**

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

    Answer: [NA]

    Justification: This work poses no such risks.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: All datasets and environments used in this work are open-source and have been appropriately cited.

    Guidelines:

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

 https://doi.org/10.52202/079017-4484

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

13. **New Assets**

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

    Answer: [Yes] .

    Justification: We will be releasing documented code on the project website along with our real-world dataset. The submission has been appropriately anonymized.

    Guidelines:

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

14. **Crowdsourcing and Research with Human Subjects**

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

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

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

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

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

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

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