BC-0: Zero-Shot Task Generalization with Robotic Imitation Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract: In this paper, we study the problem of enabling a vision-based robotic manipulation system to generalize to novel tasks, a long-standing challenge in robot learning. We approach the challenge from an imitation learning perspective, aiming to study how scaling and broadening the data collected can facilitate such generalization. To that end, we develop an interactive and flexible imitation learning system that can learn from both demonstrations and interventions and can be conditioned on different forms of information that convey the task, including pre-trained embeddings of natural language or videos of humans performing the task. When scaling data collection on a real robot to more than 100 distinct tasks, we find that this system can perform 21 *unseen* manipulation tasks with an average success rate of 44%, without any robot demonstrations for those tasks.

Keywords: Zero-Shot Imitation Learning, Multi-Task Imitation, Deep Learning

1 Introduction

One of the grand challenges in robotics is to create a general-purpose robot capable of performing a multitude of tasks in unstructured environments based on arbitrary user commands. The key challenge in this endeavour is *generalization* - the robot must handle new environments, recognize and manipulate objects it has not seen before, and understand the intent of a command it has never been asked to execute. End-to-end machine learning from pixels is a flexible choice for modeling the behavior of such generalist robots, since it has minimal assumptions about the state representation of the world. Given sufficient real-world data, these methods should in principle enable robots to generalize across new tasks, objects, and scenes without requiring hand-coded, task-specific representations. However, realizing this goal has generally remained elusive. In this paper, we study the problem of enabling a robot to generalize zero-shot to new vision-based manipulation tasks.

We study this problem using the framework of imitation learning. Prior works on imitation learning have successfully shown one-shot or zero-shot generalization to new objects [1, 2, 3, 4, 5] and to new object goal configurations [6, 7]. However, zero-shot generalization to new tasks remains a major challenge, particularly when considering vision-based manipulation tasks that cover a breadth of skills (e.g., grasping, wiping, pushing, pick-and-place) with diverse objects. Achieving such generalization depends on solving challenges relating to scaling up data collection, algorithms, and models for diverse data.

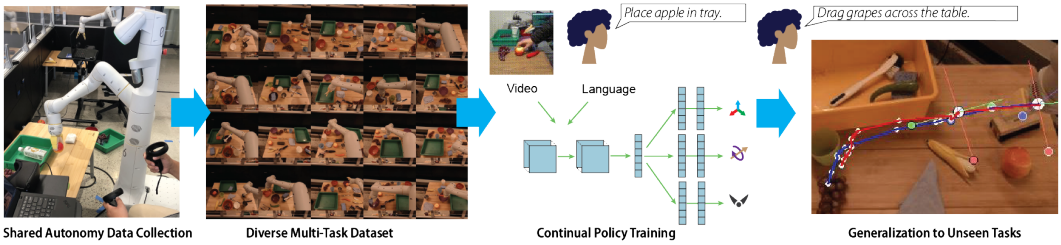

Figure 1: Overview of BC-0. We collect a large-scale dataset (25,877 episodes) of 100 diverse manipulation tasks, and train a 7-DoF multi-task policy that conditions on task language strings or human video. We show this system produces a policy that is capable of generalizing zero-shot to new unseen tasks.

Submitted to the 5th Conference on Robot Learning (CoRL 2021). Do not distribute.

We develop an interactive imitation learning system with two key properties that enable us to collect high-quality data and generalize to entirely new tasks. First, our system incorporates shared autonomy into teleoperation to allow us to collect both raw demonstration data and human interventions to correct the robot's current policy. Second, our system flexibly conditions the policy on different forms of task specification, including a language instruction or a video of a person performing the task. Unlike conditioning on a discrete one-hot task identifier [8], these continuous forms of task specification can in principle enable the robot to generalize zero-shot to new tasks by providing a language or video command of the new task at test time. Both of these properties have been explored previously, but our aim is to empirically demonstrate that these ideas continue to scale to a broad range of real-world tasks.

The main contribution of this work is a large-scale interactive imitation learning system that allows policies to solve a breadth of tasks, including tasks not seen during training. Using this system, we collect a large dataset of 100 robotic manipulation tasks, through a combination of expert teleoperation and a shared autonomy process where the human operator "coaches" the learned policy by fixing its mistakes. Across 12 robots, 7 different operators collected 25,877 robot demonstrations that totaled 125 hours of robot time, as well as 18,726 human videos of the same tasks. At test time, the system is capable of performing 21 unseen manipulation tasks in a zero-shot manner between objects that have never previously appeared together in the same scene. These closed-loop visuomotor policies perform decision-making asynchronously at 10Hz, amounting to well over 100 decisions per episode (i.e. long-horizon tasks that would be challenging for sparse RL objectives).

2 Related Work

Imitation learning has been successful in learning grasping and pick-place tasks from low-dimensional state [9, 10, 11, 12, 13, 14, 15]. Deep learning has enabled imitation learning directly from raw image observations [8, 16, 17]. In this work, we focus on enabling zero-shot generalization to new tasks in an imitation learning framework.

Multiple prior imitation learning works have achieved different forms of generalization, including one-shot generalization to novel objects [1, 2, 3, 4, 18], to novel object configurations [19], and to novel goal configurations [6, 7, 20], as well as zero-shot generalization to new objects [5], scenes [21], and goal configurations [22]. Many of these works adapt to the new scenario by conditioning on a robot demonstration [1, 2], a video of a human [3, 4], a language instruction [23, 24], or a goal image [21]. Our system flexibly conditions on either a video of a human or a language instruction, and we focus on achieving zero-shot generalization to entirely new 7-DoF manipulation tasks on a real robot, including scenarios without goal images where task-relevant objects are never encountered together in the training data.

It is standard to collect demonstrations via teleoperation [25] or kinesthetic teaching [10], and active learning methods such as DAgger [26] help reduce distribution shift for the learner. Unfortunately, DAgger and some of its variants [27, 28] are notoriously difficult to apply to robotic manipulation because they necessitate an interface where the expert must annotate the correct action when not in control of the robot policy. Inspired by recent works in autonomous driving, HG-DAgger [29] and EIL [30], our system instead only requires the expert to intervene when they believe the policy is likely to make an error and allows the expert to temporarily take full control to put the policy back on track. The resulting data collection scheme is easy to use and helps address distribution shift. Furthermore, the rate of expert interventions during data collection can be used as a live evaluation metric, which we empirically find correlates with policy success.

Beyond imitation learning, generalization has been studied in a number of other robot learning works. This includes works that generalize skills to novel objects [31, 32, 33, 34, 35], to novel environments [36], from simulation to reality [37, 38, 39, 40, 41], and to new manipulation skills and objects [42, 43, 44, 45]. We focus on the last case of generalizing to novel tasks, but unlike these prior works, we tackle a large suite of 100 challenging tasks that involve 7 DoF control at 10 Hz and involve more than 100 decisions within an episode to solve the task.

3 Problem Setup and Method Overview

An overview of our imitation learning system is shown in Figure 1. Our goal is to train a conditional policy that can interpret RGB images, denoted $s \in \mathcal{S}$, together with a task command $w \in \mathcal{W}$, which might correspond to a language string or a video of a person. The policy is a mapping from images and commands to an action space, and can be written as $\mu : \mathcal{S} \times \mathcal{W} \to \mathcal{A}$, where the action space $\mathcal{A}$

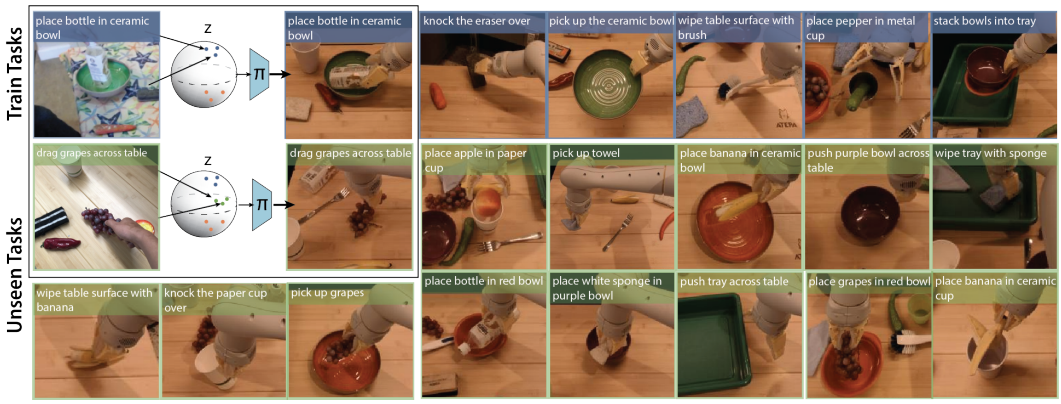

Figure 2: A subset of training tasks (top row), and a subset of held-out tasks (bottom two rows) used for evaluating zero shot task generalization. Top left: Given a pretrained task embedding computed from human videos or text, BC-0 acts as an "action decoder" for the task embedding.

consists of the 6-DoF pose of the end effector (position and orientation), as well as a 7th degree of freedom for continuous control of the parallel jaw gripper.

The policy is trained using a large-scale dataset collected via a VR-based teleoperation rig (see Figure 1, left) through a combination of direct demonstration, and human-in-the-loop shared autonomy. In the latter, trained policies are deployed on the robot, and the human operator intervenes to provide corrections when the robot makes a mistake. This procedure resembles the human-gated DAgger (HG-DAgger) algorithm [26, 29], and provides iterative improvement for the learned policy, as well as a continuous signal that can be used to track the policy's performance.

The policy architecture is divided into an encoder $q(z|w)$, which processes the command w into an embedding $z \in \mathcal{Z}$, and a control layer π, which processes (s, z) to produce the action a, i.e. $\pi : \mathcal{S} \times \mathcal{Z} \to \mathcal{A}$. This decomposition is illustrated in Figure 2, with further details in Section 5. It provides our method with the ability to incorporate auxiliary supervision, such as pretrained language embeddings, which help to structure the latent task space and facilitate generalization. In our experiments, we will show that this enables generalization to tasks that were not seen during training, including novel compositions of verbs and objects.

4 Data Collection and Workflow

In order for an imitation learning system to generalize to new tasks with zero demonstrations of said task, we must be able to easily collect a diverse dataset, provide corrective feedback, and evaluate many tasks at scale. In this section, we discuss these components of our system.

System Setup. Our teleoperation system uses an Oculus VR headset which is attached to the robot's onboard computer via USB cable and tracks two handheld controllers. The teleoperator stands behind the robot and uses the controllers to operate the robot with a line-of-sight 3rd-person view. The robot responds to the operator's movement in a 10 Hz non-realtime control loop. The relatively fast closed-loop control allows the operator to demonstrate a wide range of tasks with ease and quickly intervene if the robot is about to enter an unsafe state during autonomous execution. Further details on the user interface and data collection are in Appendices A and B.

Environment and Tasks. We place each robot in front of a table with anywhere from 6 to 15 household objects with randomized poses. We collect demonstrations and videos of humans for 100 pre-specified tasks (listed in Table 6 and Appendix J). The model is then evaluated on 29 *new* tasks using a new language description or video of that task. For the method to perform well on these held-out tasks, it must both correctly interpret the new task commands and output actions that are consistent with that task.

Shared Autonomy Data Collection. Data collection begins with an initial expert-only phase, where the human provides the demonstration of the task from start-to-finish. After an initial multi-task policy is learned from expert-only data, we continue collecting demonstrations in "shared autonomy" mode, where the current policy attempts the task while the human supervises. At any point the human may take over by gripping an "override" switch, which allows them to briefly take full control of the robot and perform necessary corrections when the policy is about to enter an unsafe state, or

if they believe the current policy will not successfully complete the task. This setup enables HG-DAgger [29], where intervention data is then aggregated with the existing data and used to re-train the policy. For the multi-task manipulation tasks, we collect 11,108 expert-only demonstrations for the initial policies, then collected an additional 14,769 HG-DAgger demonstrations covering 16 iterations of policy deployment. This gives a total of 25,877 robot demos. We find in Table 4 that when controlling for the same number of total episodes, HG-DAgger improves performance substantially.

Shared Autonomy Evaluation. When success rates are low, resources are best spent on collecting more data to improve the policy; but evaluation is also important to debug problems in the workflow. As the expected degree of generalization increases, we need more trials to evaluate the extent of policy generalization. This creates a resource trade-off: how should robot time be allocated between measuring policy success rates and collecting additional demonstrations to improve the policy? Fortunately, shared autonomy data collection confers an additional benefit: the *intervention rate*, measured as the average number of interventions required per episode, can be used as an indication for policy performance. In Figure 6, we find that the intervention rate correlates negatively with overall policy success rate.

5 Learning Algorithm

The data collection procedure above results in a large multi-task dataset. For each task i, this dataset contains expert data $(s, a) \in \mathcal{D}_e^i$, human video data $w_h \in \mathcal{D}_h^i$, and one language command w_ℓ^i. We now discuss how we use this data to train the encoder $q(z|w)$ and the control layer $\pi(a|s, z)$.

5.1 Language and Video Encoders

Our encoder $q(z|w)$ takes either a language command w_ℓ^i or a video of a human w_h as input and produces a task embedding z. If the command is a language command, we use a pretrained multi-lingual sentence encoder [46][1] as our encoder, producing a 512-dim language vector for each task. Despite the simplicity, we find that these encoders work well in our experiments.

When task commands are instead a video of a human performing the task, we use a convolutional neural network to produce z, specifically a ResNet-18 based model. Inspired by recent works [2, 3], we train this network in an end-to-end manner. We collected a dataset of 18,726 videos of humans doing each training task, in a variety of home and office locations, camera viewpoints, and object configurations. Using paired examples of a human video w_h^i and corresponding demonstration demo $\{(s, a)\}^i$, we encode the human video $z^i \sim q(\cdot \mid w_h^i)$, then pass the embedding to the control layer $\pi(a|s, z^i)$, and then backpropagate gradient of the behavior cloning loss to both the policy and encoder parameters.

Visualizations of learned embeddings in Appendix E indicate that by itself, this end-to-end approach tends to overfit to initial object scenes, learn poor embeddings, and show poor task generalization. To help align the video embeddings more semantically, we therefore further introduce an auxiliary *language regression* loss. Concretely, this auxiliary loss trains the video encoder to predict the embeddeding of the task's language command with a cosine loss. The resulting video encoder objective is as follows:

$$\min \sum_{\text{task } i} \sum_{\substack{(s,a)\sim\mathcal{D}_e^i \\ w_h\sim\mathcal{D}_h^i \bigcup \mathcal{D}_e^i}} \underbrace{-\log \pi(a|s, z^i)}_{\text{behavior cloning}} + \underbrace{D_{\cos}(z_h^i, z_\ell^i)}_{\text{language regression}}, \text{ where } \underbrace{z_h^i \sim q(\cdot|w_h)}_{\text{video encoder}}, \underbrace{z_\ell^i \sim q(\cdot|w_\ell^i)}_{\text{language encoder}} \quad (1)$$

where $D_{\cos}$ denotes the cosine distance. Since robot demos double as videos of the task, we also train encoded robot videos to match to the language vector. This language loss is critical to learning a more organized embedding space. Additional architecture and training details are in Appendix E.

5.2 Policy Training

Given a fixed task embedding, we train $\pi(a|s, z)$ via Huber loss on XYZ and axis-angle predictions, and log loss for the gripper angle. During training, images are randomly cropped, downsampled, and subjected to standard photometric augmentations. Below we describe two additional design choices that we found to be helpful. Additional training details such as learning rates, batch sizes, pseudocode, and further hyperparameters are discussed in Appendix D.

[1] Checkpoint from https://tfhub.dev/google/universal-sentence-encoder-multilingual/3

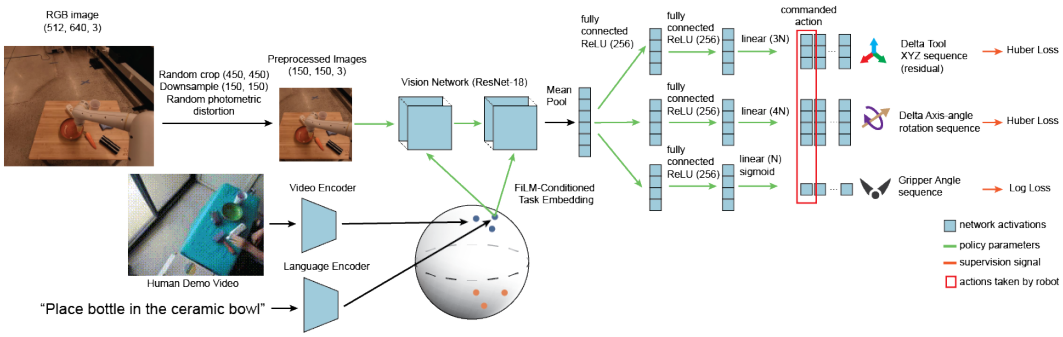

Figure 3: BC-0 Network architecture. A monocular RGB image from the head-mounted camera is passed through a ResNet18 encoder, then through a two-layer MLP to predict each action modality (delta XYZ, delta axis-angle, and gripper angle). FiLM layers [47] condition the architecture on a task embedding z computed from language w_ℓ or video w_h. The policy is not jointly trained with the embedding, so in principle any embedding space over tasks can be utilized.

Open-Loop Auxiliary Predictions. The policy predicts the action the robot would take, as well as an open-loop trajectory of the next 10 actions the policy would take if it were operating in an open-loop manner. At inference time, the policy operates closed-loop, only executing the first action based on the current image. However, the open-loop prediction confers an auxiliary training objective, and provides a way to visually inspect the quality of a closed-loop plan in an offline manner (see Figure 1, right). In pick-and-place tasks for instance, the trajectory plan can reveal if the model knows which objects it should proceed towards after it has picked up the first object.

State Differences as Actions. In standard imitation learning implementations, actions taken at demonstration-time are used directly as target labels to be predicted from states. However, cloning expert actions at 10Hz resulted in the policy learning very small actions, as well as dithering behavior. To address this, we define actions as state differences to target poses $N > 1$ steps in the future, using an adaptive algorithm to choose N based on how much the arm and gripper move. We provide ablation studies for this design choice in Section 6.3 and further details in Appendix C

5.3 Network Architecture

We model the policy using a deep neural network, shown in Figure 3. The policy network processes the camera image with a ResNet18 "torso" [48], which branches from the last mean-pool layer into multiple "action heads". Each head is a multilayer perceptron with two hidden layers of size 256 each and ReLU activations, and models part of the end-effector action, specifically the delta XYZ, delta axis-angle, and normalized gripper angle. The policy is conditioned on a 512-dim task embedding z, through FiLM layers [47]. Following Perez et al. [47], the task conditioning is linearly projected to channel-wise scales and shifts for each channel of each of the 4 ResNet blocks.

6 Experimental Results

Our experiments aim to evaluate BC-0 in large-scale imitation learning settings. We start with an initial validation of BC-0 on single-task visual imitation learning. Then, our experiments will aim to answer the following questions: (1) Can BC-0 enable zero-shot generalization to new tasks from a command in the form of language or a video of a human? (2) Is the performance of BC-0 bottlenecked by the task embedding or by the policy? (3) How important are different components of BC-0, including HG-DAgger data collection and adaptive state diffs? Additional qualitative results are on the project site: https://sites.google.com/view/bc0-anon.

6.1 BC-0 on Single-Task Imitation Learning

We first aim to verify that BC-0 can learn individual vision-based tasks before considering the more challenging multi-task setting. We choose two tasks: a *bin-emptying* task where the robot must grasp objects from a bin and drop them into an adjacent bin, and a *door opening* task where the robot must push open a door while avoiding collisions. Both tasks use the architecture in Figure 3, except that the door opening task involves predicting the forward and yaw velocity of the base instead of controlling the arm. The bin-emptying dataset has 2,759 demonstrations, while the door opening

Figure 4: Bin Emptying and Door Opening tasks are used to validate that BC-0 can achieve a high level of task success and generalize to unseen scenes in a single-task imitation learning setting.

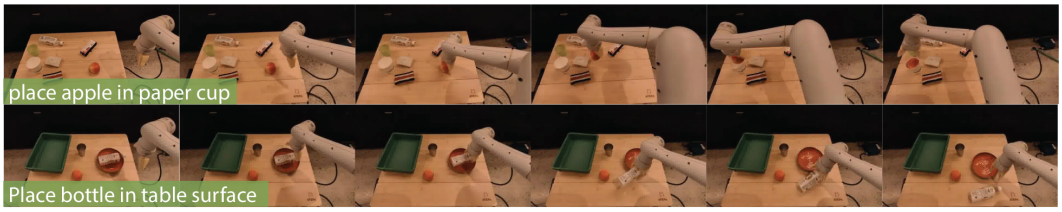

Figure 5: Qualitative examples of BC-0 successfully performing held-out tasks.

dataset has 12,000 demonstrations collected across 24 meeting rooms and 36,000 demonstrations across 5 meeting rooms in simulation. Further task and dataset details are in Appendix I.

In Table 1, we see that the BC-0 model is able to reach a pick-rate of 3.4 picks per minute, over half the speed of a human teleoperator. Further, we see that BC-0 reaches a success rate of 87% on the training door scenes and 94% on held-out door scenes. These results validate that the BC-0 model and data collect system can achieve good performance on both training and held-out scenes in the single-task setting. Additional analysis is provided in Appendix H.

Table 1: Single-task bin and door performance, average and standard deviation across runs.

Bin-Emptying	**Picks / Minute**	**# Runs**
Human Expert	6.3 (2.1)	2759
BC-0 (2759 demos)	3.4 (1.2)	9
Door Opening	**Success Rate**	**# Runs**
BC-0 (24 Train Doors)	87% (2.2)	480
BC-0 (4 Holdout Doors)	94% (2.7)	80

6.2 Evaluating Zero-Shot Task Generalization

Next, we aim to test whether BC-0 can achieve zero-shot generalization to new tasks. Demonstrations are collected across 100 different manipulation tasks, comprising two disjoint sets of objects. Using disjoint sets of objects allows us to specifically test generalization to combinations of object-object pairs and object-action pairs that are not seen together during training. For the first set of objects, demonstrations are collected across 21 different tasks, listed in Table 6, which cover a wide range of skills, from pick-and-place tasks to skills that require positioning the object in a certain way, like "stand the bottle upright". For the second set of objects, demonstrations are collected for 79 different tasks, including pick-and-place, surface wiping, and object stacking. The latter family has a smaller variety of manipulation behaviors, but is defined over a larger object set with more clutter. Object sets are shown in Appendix B and a full list of train task sentences are in Appendix J.

We evaluate BC-0 on 29 held-out tasks. Four held-out tasks are generated with objects in the 79-task family, whereas 25 tasks are generated by mixing objects between the 21-task family and 79-task family. Thus, the first 4 held-out tasks do not require cross-object set generalization, so they are easier to generalize to. Even so, we find that each of these 4 tasks are sufficiently challenging that training single-task policies on 300+ held-out demos with DAgger interventions completely fails, achieving 0% task success. This provides a degree of calibration on the difficulty of these tasks. We hypothesize that a major contributing factor to this challenge is the wide range of locations, objects, and distractors that the skills must generalize to in our settings, as well as the wide range of these factors in the training data.

In Table 2, we see that language-conditioned BC-0 is able to generalize zero-shot to both kinds of held-out tasks, averaging at 32% success and showing non-zero success on 21 held-out tasks. Among the 21 hold-out tasks with non-zero success rates, BC-0 achieves an average success of 44% when conditioned on language embeddings it has never seen. When conditioning on videos of humans, we find that generalization is much more difficult, but that BC-0 is still able to generalize to nine novel tasks with a non-zero success rate, particularly when the task does not involve novel object combinations. Qualitatively, we observe that the language-conditioned policy usually moves

Table 2: Evaluating zero-shot task generalization. The first 4 tasks only use objects from the 79-task family. The remaining tasks mix objects between the 21-task and 79-task families, requiring further generalization. Numbers in parentheses are 1 unit standard deviation. The language conditioning generalizes to several holdout tasks, whereas the video conditioning shows promise on tasks that do not mix objects between task families.

Skill	**Held-out tasks (no demos during training)**	**Lang-conditioned performance**	**Video-conditioned performance**
pick-place	'place sponge in tray'	82% (9.2)	22% (2.2)
	'place grapes in red bowl'	75% (10.8)	12% (7.8)
	'place apple in paper cup'	33% (12.2)	14% (7.8)
pick-wipe	'wipe tray with sponge'	0% (0)	28% (10.6)
pick-place	'place banana in ceramic bowl'	75% (9.7)	7.5% (4.2)
	'place bottle in red bowl'	75% (9.7)	0% (0)
	'place grapes in ceramic bowl'	70% (10.3)	0% (0)
	'place bottle in table surface'	50% (11.2)	5% (3.5)
	'place white sponge in purple bowl'	45% (11.2)	0% (0)
	'place white sponge in tray'	40% (11.0)	0% (0)
	'place apple in ceramic bowl'	20% (8.9)	0% (0)
	'place bottle in purple bowl'	20% (8.9)	0% (0)
	'place banana in ceramic cup'	0% (0)	0% (0)
	'place banana on white sponge'	0% (0)	0% (0)
	'place metal cup in red bowl'	0% (0)	0% (0)
grasp	'pick up grapes'	65% (10.7)	0% (0)
	'pick up apple'	55% (11.2)	5% (3.5)
	'pick up towel'	42.8% (18.7)	0% (0)
	'pick up pepper'	35% (10.7)	12.5% (5.2)
	'pick up bottle'	30% (10.3)	17.5% (6.0)
	'pick up the red bowl'	0% (0)	0% (0)
pick-drag	'drag grapes across the table'	14% (13.2)	0% (0)
pick-wipe	'wipe table surface with banana'	10% (6.7)	0% (0)
	'wipe tray with white sponge'	0% (0)	0% (0)
	'wipe ceramic bowl with brush'	0% (0)	0% (0)
push	'push purple bowl across the table'	30% (10.3)	0% (0)
	'push tray across the table'	25% (9.7)	0% (0)
	'push red bowl across the table'	0% (0)	0% (0)
	Holdout Task Overall	32%	4%

towards the correct objects, clearly indicating that the task embedding is reflective of the correct task, as we further illustrate in the supplementary video. The most common source of failures are "last-centimeter" errors: failing to close the gripper, failing to let go of objects, or a near miss of the target object when letting go of an object in the gripper.

Is Performance Bottlenecked on the Encoder or the Policy? Now that we see that BC-0 can generalize to a substantial number of held-out tasks to some degree, we ask whether the performance is limited more by the generalization of the encoder $q(z|w)$, the control layer $\pi(a|s, z)$, or both. To disentangle these factors, we measure the policy success rate on the training tasks conditioned in three ways: a one-hot task identifier, language embeddings of the training task commands, and video embeddings of *held-out* human videos of the training tasks. This comparison is in Table 3. The similar performance between one-hot and language suggests the latent language space is sufficient, and that language-conditioned performance on held-out tasks is bottlenecked on the control layer more than the embedding. The more significant drop in performance of video-conditioned policies suggests inferring tasks from videos is much more difficult, particularly for held-out tasks.

Table 3: Training vs. generalization performance, averaged across 21 of the training tasks and all 28 held-out tasks.

Setting	**Task Conditioning**	**Success**
Train	One-hot	42%
	Language	40%
	Video	24%
Held-Out	Language	32%
	Video	4%

Table 4: Ablation Studies. Left: Multi-task vs. single task models on the 'place the bottle in the ceramic bowl' task. Training across tasks and with adaptive state-diffs is important for good training performance. Right: DAgger comparison on 'place the bottle in the ceramic bowl' (1-Task) and the 8-Task subset from Table 6. Controlled for the same amount of data, DAgger reaches higher success numbers significantly more quickly.

Method	**1-Task**
Multi-task, language conditioned	52% (6.3)
Multi-task, one-hot conditioned	45% (5.3)
Single-task baseline (1000 demos)	5% (2.8)
Multi-task, one-hot, no adaptive state-diff	3% (2.3)

Dataset	**1-Task**	**8-Task**
100% Manual	27% (5.2)	23% (4.2)
50% Manual + 50% HG-DAgger	53% (5.8)	47% (5.2)

6.3 Ablation Studies and Comparisons

We now validate the importance of several BC-0 design decisions using the (training) 21-task family. Our first set of ablations evaluate on the "place the bottle in ceramic bowl" command, which has the most demos (1000) of any task. We first test whether multi-task training is helpful for performance: we compare the multi-task system trained on 25,877 demos across all tasks, to a single-task policy trained on just the 1000 demos for the target task. In Table 4 (left), the single-task baseline achieves just 5% success. The low number is consistent with the low single-task performance on holdout tasks from Section 6.2: collecting data over several robots and operators likely makes the task harder to learn and only when pooling much more data across many tasks does BC-0 learn to solve the task. We ablate the adaptive state diff scheme described in Section 5.3 and find that it is quite important; when naively choosing the $N = 1$ future state to compute the expert actions, the policy fits the noise and moves too slowly, resulting in state drift away from good trajectories.

We next ablate the use of HG-DAgger while keeping the total amount of data fixed. Specifically, we compare performance of policies trained using 50% expert demos and 50% HG-DAgger interventions, versus using 100% expert demos. In Table 4 (right), we find that HG-DAgger significantly improves task performance over cloning expert demonstrations on both the 'place bottle in ceramic bowl' task and 7 other training tasks. Further details on this comparison are in Appendix K. Finally, in Figure 6, we evaluate whether measuring HG-DAgger interventions can give us a live proxy of policy performance. We see that intervention frequency is inversely correlated with policy success, as measured by the fraction of successful episodes not requiring intervention. This result suggests that we can indeed use this metric with HG-DAgger for development purposes.

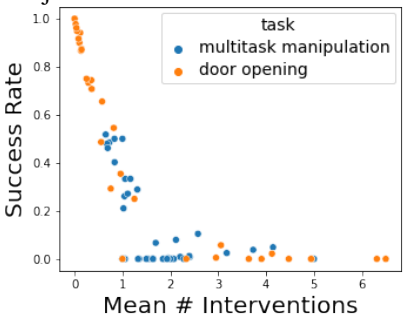

Figure 6: Mean number of interventions vs. task success rate. Each dot represents a policy evaluated during HG-DAgger data collection. There is a clear correlation between the mean number of interventions and success rate, suggesting that interventions can be used as a live proxy for performance.

7 Discussion

We presented a multi-task imitation learning system that combines flexible task embeddings with large-scale training on a 100-task demonstration dataset, enabling it to generalize to entirely new tasks that were not seen in training based on user-provided language or video commands. Our evaluation covered 29 unseen vision-based manipulation tasks with a variety of objects and scenes. Our system does have a number of limitations. First, the zero-shot performance on new tasks varies significantly. However, even for tasks that are less successful, the robot often exhibits behavior suggesting that it understands at least part of the task, reaching for the right object or performing a semantically related motion. This suggests that an exciting direction for future work is to use our policies as a general-purpose initialization for finetuning of downstream tasks, where additional training, perhaps with autonomous RL, could lead to significantly better performance. The structure of our language commands follows a simple "(verb) (noun)" structure. A direction to address this limitation is to relabel our data with a variety of human-provided annotations [24], which could enable the system to handle more variability in the language structure. Another limitation is the lower performance of the video-conditioned policy. Given the success of conditioning on a pretrained language encoder, conditioning on a pretrained video encoder trained on large-scale datasets [49] could be equally effective, providing a way for robot learning to leverage non-robotics datasets from the wider machine learning community.

References

[1] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine. One-shot visual imitation learning via meta-learning. In *Conference on Robot Learning*, pages 357–368. PMLR, 2017.

[2] S. James, M. Bloesch, and A. J. Davison. Task-embedded control networks for few-shot imitation learning. In *Conference on Robot Learning*, pages 783–795. PMLR, 2018.

[3] T. Yu, C. Finn, A. Xie, S. Dasari, T. Zhang, P. Abbeel, and S. Levine. One-shot imitation from observing humans via domain-adaptive meta-learning. *Robotics: Science and Systems*, 2018.

[4] A. Bonardi, S. James, and A. J. Davison. Learning one-shot imitation from humans without humans. *IEEE Robotics and Automation Letters*, 5(2):3533–3539, 2020.

[5] S. Young, D. Gandhi, S. Tulsiani, A. Gupta, P. Abbeel, and L. Pinto. Visual imitation made easy. *arXiv e-prints*, pages arXiv–2008, 2020.

[6] Y. Duan, M. Andrychowicz, B. C. Stadie, J. Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba. One-shot imitation learning. *arXiv preprint arXiv:1703.07326*, 2017.

[7] S. Dasari and A. Gupta. Transformers for one-shot visual imitation. *Conference on Robot Learning (CoRL)*, 2020.

[8] R. Rahmatizadeh, P. Abolghasemi, L. Bölöni, and S. Levine. 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)*, pages 3758–3765. IEEE, 2018.

[9] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

[10] S. Khansari-Zadeh and A. Billard. Learning stable non-linear dynamical systems with gaussian mixture models. *IEEE Transaction on Robotics*, 27(5):943–957, 2011.

[11] A. Billard, Y. Epars, S. Calinon, S. Schaal, and G. Cheng. Discovering optimal imitation strategies. *Robotics and autonomous systems*, 47(2-3):69–77, 2004.

[12] S. Schaal, J. Peters, J. Nakanishi, and A. Ijspeert. Learning movement primitives. In *Robotics research. the eleventh international symposium*, pages 561–572. Springer, 2005.

[13] R. Chalodhorn, D. B. Grimes, K. Grochow, and R. P. Rao. Learning to walk through imitation. In *IJCAI*, volume 7, pages 2084–2090, 2007.

[14] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal. Learning and generalization of motor skills by learning from demonstration. In *2009 IEEE International Conference on Robotics and Automation*, pages 763–768. IEEE, 2009.

[15] K. Mülling, J. Kober, O. Kroemer, and J. Peters. Learning to select and generalize striking movements in robot table tennis. *The International Journal of Robotics Research*, 32(3):263–279, 2013.

[16] D. A. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. Technical report, Carnegie-Mellon University, 1989.

[17] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5628–5635. IEEE, 2018.

[18] A. Zhou, E. Jang, D. Kappler, A. Herzog, M. Khansari, P. Wohlhart, Y. Bai, M. Kalakrishnan, S. Levine, and C. Finn. Watch, try, learn: Meta-learning from demonstrations and reward. *International Conference on Learning Representations (ICLR)*, 2020.

[19] T. L. Paine, S. G. Colmenarejo, Z. Wang, S. Reed, Y. Aytar, T. Pfaff, M. W. Hoffman, G. Barth-Maron, S. Cabi, D. Budden, et al. One-shot high-fidelity imitation: Training large-scale deep nets with rl. *arXiv preprint arXiv:1810.05017*, 2018.

[20] D.-A. Huang, S. Nair, D. Xu, Y. Zhu, A. Garg, L. Fei-Fei, S. Savarese, and J. C. Niebles. Neural task graphs: Generalizing to unseen tasks from a single video demonstration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8565–8574, 2019.

[21] D. Pathak, P. Mahmoudieh, G. Luo, P. Agrawal, D. Chen, Y. Shentu, E. Shelhamer, J. Malik, A. A. Efros, and T. Darrell. Zero-shot visual imitation. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 2050–2053, 2018.

[22] P. Goyal, R. J. Mooney, and S. Niekum. Zero-shot task adaptation using natural language. *arXiv preprint arXiv:2106.02972*, 2021.

[23] S. Stepputtis, J. Campbell, M. Phielipp, S. Lee, C. Baral, and H. B. Amor. Language-conditioned imitation learning for robot manipulation tasks. *arXiv preprint arXiv:2010.12083*, 2020.

[24] C. Lynch and P. Sermanet. Grounding language in play. *arXiv preprint arXiv:2005.07648*, 2020.

[25] S. Calinon, P. Evrard, E. Gribovskaya, A. Billard, and A. Kheddar. Learning collaborative manipulation tasks by demonstration using a haptic interface. In *2009 International Conference on Advanced Robotics*, pages 1–6. IEEE, 2009.

[26] 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.

[27] S. Ross and J. A. Bagnell. Reinforcement and imitation learning via interactive no-regret learning. *arXiv preprint arXiv:1406.5979*, 2014.

[28] M. Laskey, J. Lee, R. Fox, A. Dragan, and K. Goldberg. Dart: Noise injection for robust imitation learning. In *Conference on robot learning*, pages 143–156. PMLR, 2017.

[29] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer. Hg-dagger: Interactive imitation learning with human experts. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8077–8083. IEEE, 2019.

[30] J. Spencer, S. Choudhury, M. Barnes, M. Schmittle, M. Chiang, P. Ramadge, and S. Srinivasa. Learning from interventions: Human-robot interaction as both explicit and implicit feedback. 07 2020. doi:10.15607/RSS.2020.XVI.055.

[31] M. Khansari, D. Kappler, J. Luo, J. Bingham, and M. Kalakrishnan. Action image representation: Learning scalable deep grasping policies with zero real world data. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3597–3603, 2020.

[32] 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.

[33] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *arXiv preprint arXiv:1703.09312*, 2017.

[34] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pages 651–673. PMLR, 2018.

[35] J. Hatori, Y. Kikuchi, S. Kobayashi, K. Takahashi, Y. Tsuboi, Y. Unno, W. Ko, and J. Tan. Interactively picking real-world objects with unconstrained spoken language instructions. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3774–3781. IEEE, 2018.

[36] A. Gupta, A. Murali, D. Gandhi, and L. Pinto. Robot learning in homes: Improving generalization and reducing dataset bias. *arXiv preprint arXiv:1807.07049*, 2018.

[37] F. Sadeghi, A. Toshev, E. Jang, and S. Levine. Sim2real viewpoint invariant visual servoing by recurrent control. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4691–4699, 2018.

[38] J. Tobin, L. Biewald, R. Duan, M. Andrychowicz, A. Handa, V. Kumar, B. McGrew, A. Ray, J. Schneider, P. Welinder, et al. Domain randomization and generative models for robotic grasping. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3482–3489. IEEE, 2018.

[39] B. Mehta, M. Diaz, F. Golemo, C. J. Pal, and L. Paull. Active domain randomization. In *Conference on Robot Learning*, pages 1162–1176. PMLR, 2020.

[40] S. James, P. Wohlhart, M. Kalakrishnan, D. Kalashnikov, A. Irpan, J. Ibarz, S. Levine, R. Hadsell, and K. Bousmalis. Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12627–12637, 2019.

[41] J. Zhang, L. Tai, P. Yun, Y. Xiong, M. Liu, J. Boedecker, and W. Burgard. Vr-goggles for robots: Real-to-sim domain adaptation for visual control. *IEEE Robotics and Automation Letters*, 4(2):1148–1155, 2019.

[42] C. Finn and S. Levine. Deep visual foresight for planning robot motion. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2786–2793. IEEE, 2017.

[43] S. Dasari, F. Ebert, S. Tian, S. Nair, B. Bucher, K. Schmeckpeper, S. Singh, S. Levine, and C. Finn. Robonet: Large-scale multi-robot learning. *arXiv preprint arXiv:1910.11215*, 2019.

[44] Y. Chebotar, K. Hausman, Y. Lu, T. Xiao, D. Kalashnikov, J. Varley, A. Irpan, B. Eysenbach, R. Julian, C. Finn, et al. Actionable models: Unsupervised offline reinforcement learning of robotic skills. *arXiv preprint arXiv:2104.07749*, 2021.

[45] D. Kalashnikov, J. Varley, Y. Chebotar, B. Swanson, R. Jonschkowski, C. Finn, S. Levine, and K. Hausman. Mt-opt: Continuous multi-task robotic reinforcement learning at scale. *arXiv preprint arXiv:2104.08212*, 2021.

[46] Y. Yang, D. Cer, A. Ahmad, M. Guo, J. Law, N. Constant, G. H. Abrego, S. Yuan, C. Tar, Y.-H. Sung, et al. Multilingual universal sentence encoder for semantic retrieval. *arXiv preprint arXiv:1907.04307*, 2019.

[47] 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.

[48] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[49] B. Fernando, H. Bilen, E. Gavves, and S. Gould. Self-supervised video representation learning with odd-one-out networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3636–3645, 2017.

[50] A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[51] D. Ho, K. Rao, Z. Xu, E. Jang, M. Khansari, and Y. Bai. Retinagan: An object-aware approach to sim-to-real transfer. In *2021 International Conference on Robotics and Automation (ICRA)*, 2021.