2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

IRIS: Implicit Reinforcement without Interaction at Scale for Learning Control from Offline Robot Manipulation Data

Ajay Mandlekar[1,2] Fabio Ramos[2,3] Byron Boots[2,4] Silvio Savarese[1] Li Fei-Fei[1] Animesh Garg[2,5] Dieter Fox[2,4]

Abstract— **Learning from offline task demonstrations is a problem of great interest in robotics. For simple short-horizon manipulation tasks with modest variation in task instances, offline learning from a small set of demonstrations can produce controllers that successfully solve the task. However, leveraging a fixed batch of data can be problematic for larger datasets and longer-horizon tasks with greater variations. The data can exhibit substantial diversity and consist of suboptimal solution approaches. In this paper, we propose Implicit Reinforcement without Interaction at Scale (IRIS), a novel framework for learning from large-scale demonstration datasets. IRIS factorizes the control problem into a goal-conditioned low-level controller that imitates short demonstration sequences and a high-level goal selection mechanism that sets goals for the low-level and selectively combines parts of suboptimal solutions leading to more successful task completions. We evaluate IRIS across three datasets, including the RoboTurk Cans dataset collected by humans via crowdsourcing, and show that performant policies can be learned from purely offline learning. Additional results at `https://sites.google.com/stanford.edu/iris/`.**

I. Introduction

Recent research has successfully leveraged Reinforcement Learning (RL) for short-horizon robotic manipulation tasks, such as pushing and grasping objects [10, 22, 37]. However, RL algorithms face the burden of efficient exploration in large state and action spaces, and consequently need large amounts of environment interaction to successfully learn policies. Furthermore, leveraging RL for policy learning requires specifying a task-specific reward function that is often carefully shaped and crafted to assist in exploration.

An appealing alternative to learning policies from scratch is to bring policy learning closer to the setting of supervised learning by leveraging prior experience. In Imitation Learning (IL), expert demonstrations are used to guide policy learning. The demonstrated data can be used in lieu of a reward function and also lessen the burden of exploration for the agent, ameliorating some of the aforementioned issues. However, IL has primarily been applied to small scale datasets collected by one decision maker. In order to truly reap the benefits of supervised learning, it is useful to consider how large-scale, diverse supervision can be used for task learning.

Large-scale human supervision has accelerated progress in computer vision and natural language processing [6, 32], but policy learning has witnessed no such success. The advent of supervision mechanisms that allow for the collection of thousands of task demonstrations in a matter of days [26] motivates the following question: does a policy learning algorithm necessarily need to *interact* with the system to learn a policy, or can a robust and performant policy be learned purely from external experiences provided in the form of a dataset? For example, consider a pick-and-place task where a robot has to pick up a soda can and place it on a shelf. We have access to a large set of task demonstrations collected via human supervision where the soda can was placed in several initial poses and people controlled the arm to demonstrate many different approaches for grasping the can and placing it on the shelf. We would like to use this dataset to train a policy that can successfully solve the task.

In order to leverage large datasets for policy learning, we argue that it is important to develop methods that are tolerant to datasets that are both suboptimal and diverse, since large-scale human supervision is likely to produce data that is highly varied in terms of both quality and task solution approaches. For example, some approaches for moving to the can and grasping it can be more efficient than others, and there are many valid ways to pick the can up. By contrast, conventional imitation learning methods assume that demonstration data is near-optimal and unimodal, and most methods start to deteriorate significantly when expert demonstrations are of lower quality, or when multiple solutions are demonstrated.

To tackle this challenge, we present Implicit Reinforcement without Interaction at Scale (IRIS), a novel framework that addresses the problem of offline policy learning from a large set of diverse and suboptimal demonstrations.

Summary of Contributions:

1) We present Implicit Reinforcement without Interaction at Scale (IRIS), a framework that enables offline learning from a large set of diverse and suboptimal demonstrations by selectively imitating local sequences from the dataset.
2) We evaluate IRIS across a pedagogical dataset, a highly suboptimal dataset, and a crowdsourced dataset, and only assume access to sparse task completion rewards that occur at the end of each demonstration.
3) Empirically, our experiments demonstrate that IRIS is able to leverage large-scale task demonstrations that exhibit suboptimality and diversity, and significantly outperforms other imitation learning and batch reinforcement learning baselines.

II. Related work

Imitation Learning and Learning from Demonstration: Imitation learning guides policy learning by leveraging a

[1] Stanford Vision & Learning Lab, [2] NVIDIA, [3] University of Sydney, [4] University of Washington, [5] University of Toronto.

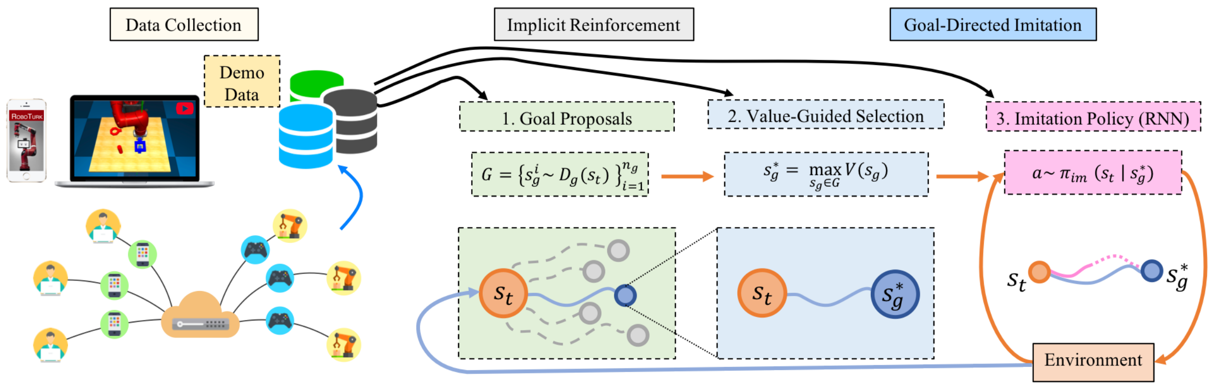
Data Collection
Implicit Reinforcement
Goal-Directed Imitation
Demo Data
1. Goal Proposals
2. Value-Guided Selection
3. Imitation Policy (RNN)
$G = \{s_g^i \sim D_g(s_t)\}_{i=1}^{n_g}$
$s_g^* = \max_{s_g \in G} V(s_g)$
$a \sim \pi_{im}(s_t \mid s_g^*)$
s_t
s_g^*
Environment

Fig. 1: **Overview** IRIS learns policies from large quantities of demonstration data without environment interaction during learning. It trains a goal-conditioned low-level controller to reproduce short demonstration sequences and a high-level goal selection mechanism consisting of a goal proposal network and a value network. At test-time, a set of goals is proposed by a generative model and selected by the value function, and this is set as the target for low-level imitation. Both high and low levels are run in closed-loop with appropriate rates.

reference set of expert demonstrations. Imitation learning methods are either offline, such as Behavioral Cloning [30, 34, 35], or online, such as Inverse Reinforcement Learning (IRL) [1, 20]. Offline methods are sensitive to the quantity of demonstration data and can suffer from covariate shift, since no additional data is collected by the agent, while online methods require additional interaction for policy learning. Furthermore, most imitation learning approaches are sensitive to the quality of expert demonstrations since they assume that the data is near-optimal.

Imitation and Reinforcement Learning from Suboptimal Demonstrations: Recent work has tried to leverage off-policy deep reinforcement learning in conjunction with a set of demonstrations to account for suboptimal data and learn policies that outperform the demonstrations [12, 14, 28, 36, 38]. However, such approaches still require significant interaction to learn policies. Furthermore, off-policy deep RL can be unstable due to the compounding effects of bootstrapping value learning and function approximation [2, 5, 11, 33]. Other methods use Batch RL to try and leverage arbitrary off-policy data for policy learning without collecting additional experience [3, 11, 16, 21, 23]. While recent efforts have produced successful continuous control policies for locomotion domains [11, 21], neither robot manipulation nor diverse demonstration data have been considered.

Goal-directed Reinforcement and Imitation Learning: Recent work has extended reinforcement learning [4, 27, 31] and imitation learning [7, 24] to condition on goal observations, enabling improved sample efficiency. HIRO [27] decomposes policy learning into a high-level policy that outputs goal observations and a low-level policy that conditions on goals and tries to achieve them. While this is similar to the architecture of IRIS, our focus is on offline learning from fixed data, and our low-level policy is trained with a supervised loss similar to [24] instead of using an off-policy RL update, which can be unsuitable for offline learning [11].

Large-Scale Data Collection in Robotics: Self-Supervised Learning has been employed to collect and learn from large amounts of data for tasks such as grasping in both simulated [13, 18, 25] and physical settings [17, 22, 29]. These methods collected hundreds of hours of robot interaction, although most of the interactions were not successful. By contrast, RoboTurk [26] is a platform that has been leveraged to collect large-scale datasets in simulation via crowdsourced human supervision, resulting in datasets with several successful demonstrations. We show in our experiments that IRIS can leverage such sources of demonstrations for successful policy learning without collecting additional samples of experience.

III. Preliminaries

Every robot manipulation task can be formulated as a sequential decision making problem. Consider an infinite-horizon discrete-time Markov Decision Process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, R, \gamma, \rho_0)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{T}(\cdot|s,a)$, is the state transition distribution, $R(s,a,s')$ is the reward function, $\gamma \in [0,1)$ is the discount factor, and $\rho_0(\cdot)$ is the initial state distribution. At every step, an agent observes s_t, uses a policy π to choose an action $a_t = \pi(s_t)$, and observes the next state $s_{t+1} \sim \mathcal{T}(\cdot|s_t,a_t)$ and reward $r_t = R(s_t,a_t,s_{t+1})$. The goal in reinforcement learning is to learn an policy π that maximizes the expected return $\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t,a_t,s_{t+1})]$.

To use this formulation for robotic task learning, we augment this MDP with a set of absorbing goal states $\mathcal{G} \subset \mathcal{S}$, where each goal state $s_g \in \mathcal{G}$ corresponds to a state of the world in which the task is considered to be solved. Similarly, every state $s_0 \sim \rho_0(\cdot)$ corresponds to a new task instance. To measure task success, we define a sparse reward function $R(s,a,s') = \mathbb{1}[s' \in \mathcal{G}]$. Consequently, maximizing expected returns corresponds to solving a task quickly and consistently. Next, we formalize the structure of the datasets we aim to leverage for task learning.

Definition 3.1 (Goal-Reaching Trajectories) Let $\tau = (s_0, a_0, r_0, s_1, ..., s_T)$ be a T-length trajectory in the MDP, where $s_0 \sim \rho_0(\cdot)$ is an initial state from the MDP with rewards $r_t = R(s_t,a_t,s_{t+1})$, and states $s_{t+1} \sim \mathcal{T}(\cdot|s_t,a_t)$ produced by the MDP given the actions $a_0, a_1, ..., a_{T-1}$. This trajectory is *goal-reaching* if the last state is a goal state, $s_T \in \mathcal{G}$.

In our setting, we assume access to a dataset $\mathcal{D}$ of N goal-reaching trajectories that has been collected by a set of policies. Our goal is to leverage this large batch of goal-reaching trajectories to learn a policy that maximizes task

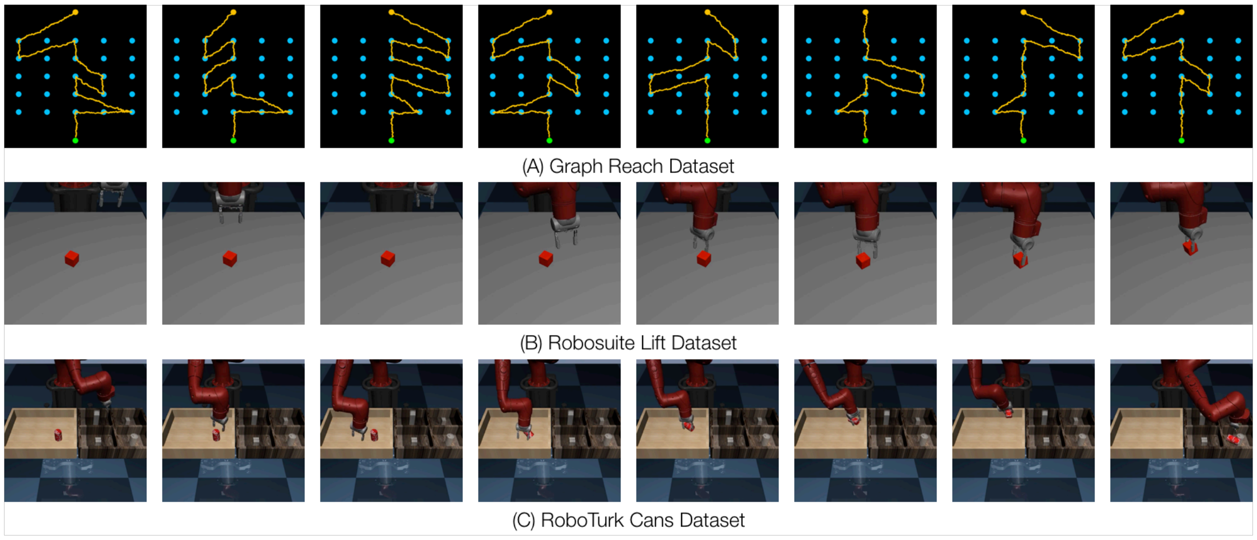

Fig. 2: **Tasks and Datasets:** The Graph Reach dataset (top) consists of demonstrated paths from the start location at the top to the goal location at the bottom. Graph nodes are sampled during each demonstration to form a connected path. The Robosuite Lift dataset (middle) was collected by one human teleoperating the robot. The human intentionally took suboptimal approaches - for example, in the demonstration shown above, the robot moves close to the cube, then far away, and then fumbles with the cube before successfully lifting it. The RoboTurk Cans dataset [26] (bottom) was collected by several humans through crowdsourcing, leading to diverse demonstrated solutions. In the above example, the person chose to knock the can over in order to pick it up.

returns. Importantly, the method cannot collect additional samples of experience in the MDP. Next, we outline some dataset properties that makes learning in this setting challenging.

Suboptimal Data: There are no guarantees placed on the quality of data-generating policies - each trajectory may take longer than necessary to solve the task. Equivalently, for a given trajectory in the dataset $\tau = (s_0, a_0, r_0, \dots, s_T) \in \mathcal{D}$ it is possible that $\sum_{t=0}^{T-1} \gamma^t r_t + \frac{\gamma^T}{1-\gamma} < V^*(s_0)$, so the task return of the demonstrated trajectory is worse than that of the optimal policy. Thus, this is different from the standard setting of imitation learning - the learned policy should not seek to imitate all demonstrated data due to variations in data quality.

Multimodal Data: Since many trajectories are in the dataset and multiple policies were used for generation, data can exhibit multimodality in how task instances are solved. For example, a soda can can be grasped from the top, or knocked down and then picked up on its side.

IV. IRIS: Implicit Reinforcement without Interaction at Scale

A. Overview

We first provide an overview of IRIS and motivate each component by how it is used at test-time. We split the decision making process into a high-level mechanism that sets goal states for a low-level controller to try and reach. At a state s_t, the high-level mechanism selects a new goal state s_g that is held constant for the next T timesteps. Then, the low-level controller is conditioned on s_g, and is given T timesteps to try and reach that state in a closed-loop fashion. Then, control is returned to the high-level and the process repeats.

We further break the high-level mechanism into 2 parts. The first part is a conditional Variational Autoencoder (cVAE) [19] that tries to model the full distribution of states $p(s_{t+T}|s_t)$ that are T timesteps away from a given state s_t. It is used to sample a set of goal proposals. The second part is a value function $V(s)$ that is used to select the most promising goal proposal. The low-level controller is a Recurrent Neural Network (RNN) that outputs an action a_t at each timestep, given a current observation s_t and goal s_g.

Together, these components allow for *selective imitation* of local sequences in the dataset. The complete training loop is provided in Algorithm 1. We next describe how each component is supervised.

B. Low-Level Goal-Conditioned Imitation Controller

The low-level goal-conditioned controller is a goal-conditioned RNN $\pi_\theta(s|g)$ (similar to [24]) trained on trajectory sequences of length T. Consecutive state-action sequences $(s_t, a_t, \dots, s_{t+T-1}, a_{t+T-1}, s_{t+T})$ are sampled from trajectories in the dataset. The last observation in each sequence, s_{t+T} is treated as a goal that the RNN should try to reach, and the RNN is trained to output the action sequence $a_t, a_{t+1}, \dots, a_{t+T-1}$ from the state sequence $s_t, s_{t+1}, \dots, s_{t+T-1}$ and the goal $s_g = s_{t+T}$ (lines 4-5 in Algorithm 1). The loss function for the RNN is a simple Behavioral Cloning loss $\mathcal{L}_\theta(a_{t:t+T}, s_{t:t+T}) = \sum_{k=t}^{t+T-1} ||a_k - \pi_\theta(s_k|s_g)||_2^2$. By learning to copy the action sequence that resulted in a particular observation, the RNN performs unimodal imitation over short demonstration sequences to reach different goals.

C. High-Level Goal Selection Mechanism

The high-level goal selection mechanism chooses goal states for the low-level to try and reach (similar to [27]). The goal selection mechanism has two components: (1) a cVAE $(E_\phi(s_g, s), D_\phi(z, s))$ to propose goal states at a particular state and (2) a value function $V(s_g)$ that models the expected return of goal states.

The cVAE is a conditional generative model that is trained on pairs of current and future observations (s_t, s_{t+T}) sampled

Algorithm 1 IRIS: Train Loop

```
Require:
    $\pi_\theta(s\,|\,s_g)$, $\{E_\phi(s_g,s), D_\phi(z,s)\}$, $Q_\psi(s,a)$, $\{E_\omega(a,s), D_\omega(z,s)\}$    ▷ Policy, Goal cVAE, Value Network, Action cVAE
 1: for $i = 1,2,...,n_{\text{iter}}$ do
 2:     $(s_t, a_t, r_t, s_{t+1}, ..., s_{t+T-1}, a_{t+T-1}, r_{t+T-1}, s_{t+T}) \sim \mathcal{D}$    ▷ Sample $T$-length sequence from the dataset
 3:     $s_g \leftarrow s_{t+T}$    ▷ Treat last observation as goal
 4:     $\hat{a}_t, \hat{a}_{t+1}, ..., \hat{a}_{t+T-1} \leftarrow \pi_\theta(s_{t:t+T-1}\,|\,s_g)$    ▷ Goal-conditioned action sequence prediction from RNN Policy
 5:     $\theta \leftarrow \arg\min_\theta \sum_{t'=t}^{t+T-1} ||a_{t'} - \hat{a}_{t'}||_2^2$    ▷ Update policy with imitation loss
 6:     $\mu_g, \sigma_g = E_\phi(s_g, s_t)$, $z \sim \mathcal{N}(\mu_g, \sigma_g)$
 7:     $\phi \leftarrow \arg\min_\phi ||s_g - D_\phi(z, s_t)||_2^2 + \beta_g KL(\mathcal{N}(\mu_g, \sigma_g)||\mathcal{N}(0,1))$    ▷ Train Goal cVAE to predict goals
 8:     $\mu_a, \sigma_a = E_\omega(a_{t+T-1}, s_{t+T-1})$, $z \sim \mathcal{N}(\mu_a, \sigma_a)$
 9:     $\omega \leftarrow \arg\min_\omega ||a_{t+T-1} - D_\omega(z, s_{t+T-1})||_2^2 + \beta_a KL(\mathcal{N}(\mu_a, \sigma_a)||\mathcal{N}(0,1))$    ▷ Train Action cVAE on last action
10:     $A \leftarrow \{a_i \sim D_\omega(s_g)\}_{i=1}^M$
11:     $\bar{V} = r_{t+T-1} + \gamma \max_{a_i \in A} Q'_\psi(s_g, a_i)$    ▷ Set target value for value update
12:     $\psi \leftarrow \arg\min_\psi (\bar{V} - Q_\psi(s_{t+T-1}, a_{t+T-1}))^2$    ▷ Update value network
13: end for
```

from trajectories in the dataset (lines 5-7 in Algorithm 1). An encoder maps a current and future observation to the parameters of a latent Gaussian distribution $\mu_g, \sigma_g = E_\phi(s_{t+T}, s_t)$ and the decoder is trained to reconstruct the future observation from the current observation and a latent sampled from the encoder distribution $\hat{s}_{t+T} = D_\phi(z, s_t)$, $z \sim \mathcal{N}(\mu_g, \sigma_G)$. The encoder distribution is regularized with a KL-loss $KL(\mathcal{N}(\mu_g, \sigma_g)||\mathcal{N}(0,1))$ with weight β_g [15] to encourage the encoder distribution to match a prior latent distribution $p(z) = \mathcal{N}(0,1)$ so that at test-time, the decoder can be used as a conditional generative model by sampling latents $z \sim \mathcal{N}(0,1)$ and passing them through the decoder.

The value function consists of a state-action value function $Q_\psi(s,a)$ trained using a simple variant of Batch Constrained Q-Learning (BCQ) [11] (lines 8-12 in Algorithm 1). The loss function for the value function is a modified version of the BCQ update, which maintains a cVAE $(E_\omega(a,s), D_\omega(z,s))$ to model a state-conditional action distribution $p(a|s)$ over the dataset, and a Q-network $Q_\psi(s,a)$ trained with a temporal difference loss, $\mathcal{L}_\psi(s,a,r,s') = (Q_\psi(s,a) - Q_{\text{target}})^2$. The target value is computed by considering a set of action proposals from the cVAE $A = \{D_\omega(z,s)\,|\,z \sim \mathcal{N}(0,1)\}_{i=1}^M$ and maximizing the Q-network over the set of actions, $Q_{\text{target}} = r + \gamma \max_{a_i \in A} Q'_\psi(s', a_i)$.

V. IRIS: Challenges of Purely Offline Data

In this section we elaborate on different properties of the method and how it addresses the challenges in our datasets.

Learning from diverse solution approaches: The goal-conditioned controller is trained to condition on future goal observations at a fine temporal resolution and produce unimodal action sequences. Consequently, it is not concerned with modeling diversity, but rather reproduces small action sequences in the dataset to move from one state to another. Meanwhile, the generative model in the goal selection mechanism proposes potential future observations that are reachable from the current observation - this explicitly models the diversity of solution approaches. In this way, IRIS decouples the problem into reproducing specific, unimodal sequences (policy learning) and modeling state trajectories that encapsulate different solution approaches (diversity), allowing for *selective imitation*.

Learning from suboptimal data: The low-level goal-conditioned controller operates for a small number of timesteps, so it has no need to account for suboptimal actions. This is because if the goal is to reach a state s_2 from s_1, and T is sufficiently small, then a policy would only be able to improve by reaching s_2 in less than T steps, which is a negligible improvement for small values of T. By contrast, the value learning component of the goal selection mechanism explicitly accounts for suboptimal solution approaches by evaluating the expected task returns of each goal and selecting the goal with the highest return.

Learning from off-policy datasets: Policy learning from arbitrary off-policy data can be challenging [11, 21]. Following prior work, IRIS deals with this issue by constraining learning to occur within the distribution of training data. The goal-conditioned controller directly imitates sequences from the training data, and the generative goal model is also trained to propose goal observations from the training data. Finally, the value learning component of the goal selection mechanism mitigates extrapolation error by making sure that the Q-network is only queried on state-action pairs that lie within the training distribution [11].

VI. Experimental Setup

A. Tasks and Datasets

Graph Reach - A Pedagogical Example: We constructed a simple task in a 2D navigation domain where the agent begins each episode at a start location and must navigate to a goal. The start and goal locations are fixed across all episodes. We generate a large, varied dataset by leveraging a 5x5 grid of points to sample random paths from the start location to the goal, and collecting demonstration trajectories by playing noisy, random magnitude actions to move along sampled random paths. Demonstration paths that deviate from the central path are made to take longer detours before

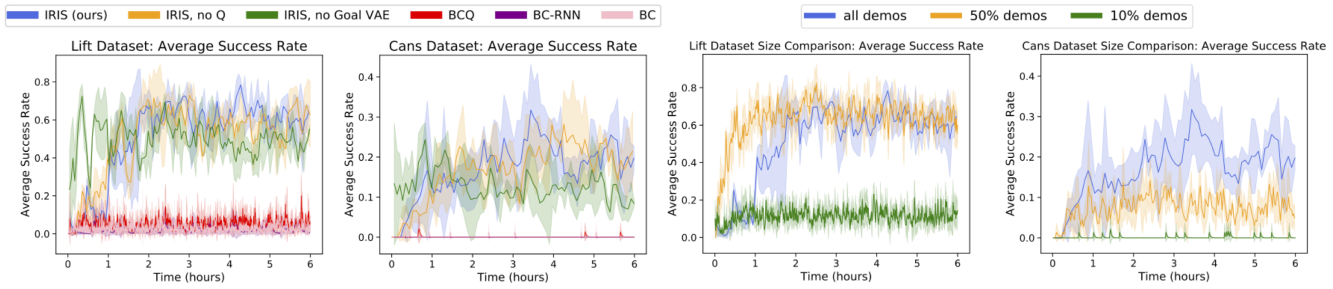

Fig. 3: **Manipulation Results:** We present a comparison of IRIS against several baselines on the Robosuite Lift and RoboTurk Cans datasets (left two plots). There is a stark contrast in performance between variants of IRIS and the baseline models, which suggests that *goal-conditioned imitation* is critical for good performance. We also perform a dataset size comparison (right two plots) to understand how the performance of IRIS is affected by different quantities of data.

TABLE I: **Performance Comparison:** We present a comparison of the best performing models for our method and baselines. Evaluations occurred on model checkpoints once per hour over 100 randomized task instances. We report the best task success rate, average rollout length (among successful rollouts), and discounted task return per training run across three random seeds. Most models are able to decrease or maintain average rollout lengths among successful rollouts compared to the original dataset of trajectories.

	Graph Reach			Robosuite Lift			RoboTurk Cans			RoboTurk Cans Image		
Model	Success Rate (%)	Rollout Length	Task Return	Success Rate (%)	Rollout Length	Task Return	Success Rate (%)	Rollout Length	Task Return	Success Rate (%)	Rollout Length	Task Return
BC	100 ± 0	2750 ± 346	67.4 ± 20.0	13.7 ± 7.36	404 ± 100	96.8 ± 59.0	0.00 ± 0.00	-	0.00 ± 0.00	13.3 ± 4.04	946 ± 70.9	55.9 ± 17.5
BC-RNN	100 ± 0	2918 ± 36.1	54.0 ± 1.93	16.7 ± 10.6	401 ± 114	117 ± 75.5	0.33 ± 0.47	166 ± 235	2.02 ± 2.86	28.3 ± 1.53	635 ± 71.5	157 ± 14.8
BCQ	100 ± 0	2077 ± 162	127 ± 19.3	18.0 ± 13.5	360 ± 65.0	132 ± 106	0.00 ± 0.00	-	0.00 ± 0.00	9.67 ± 3.06	706 ± 156	52.2 ± 19.3
IRIS, no Goal VAE	**100 ± 0**	**1895 ± 131**	**151 ± 18.9**	73.0 ± 5.35	533 ± 38.7	432 ± 47.9	21.0 ± 3.27	593 ± 15.6	117 ± 19.9	38.7 ± 6.66	632 ± 28.2	213 ± 35.1
IRIS, no Q	100 ± 0	2285 ± 227	107 ± 24.8	74.3 ± 14.9	513 ± 18.1	447 ± 89.4	**30.7 ± 3.68**	**618 ± 38.5**	**168 ± 23.8**	**42.7 ± 5.03**	**661 ± 8.92**	**230 ± 30.2**
IRIS (Full Model)	100 ± 0	2264 ± 171	106 ± 18.4	**81.3 ± 6.60**	**523 ± 29.0**	**486 ± 49.7**	**28.3 ± 0.94**	**569 ± 11.5**	**163 ± 5.68**	**42.3 ± 1.15**	**625 ± 34.6**	**236 ± 12.3**
Dataset (Oracle)	100 ± 0	3844 ± 644	27.0 ± 22.2	100 ± 0	622 ± 192	546 ± 92.7	100 ± 0	590 ± 84.0	566 ± 48.6	100 ± 0	590 ± 84.0	566 ± 48.6

joining the central path again (see Fig. 2). Several varied demonstrated paths are available in the dataset, and only certain parts of each path should be imitated to yield optimal performance. The algorithm needs to be able to recover a policy that follows the straight line path from the start to the goal by choosing to imitate pieces of the demonstrations in the dataset (for example the first, second, and third part of the three paths respectively, in the top right 3 images of Fig. 2). The dataset contains 250 demonstrations with an average completion time of 3844 timesteps.

Robosuite Lift - Suboptimal Demonstrations from a Human: We collected human demonstrations from a single human using RoboTurk [26] on the Robosuite Lifting task [9]. The goal is to actuate the Sawyer robot arm to grasp and lift the cube on the table. The demonstrator lifted the cube with a consistent grasping strategy, but took their time to grasp the cube, often moving the arm to the cube and then back, or actuating the arm from side to side near the cube, as shown in Fig. 2. This was done intentionally to ensure that there would be several state-action pairs in the dataset with little value. Algorithms need to avoid being misled by the suboptimal paths taken by the demonstrator. The dataset contains 137 demonstrations with an average completion time of 622 timesteps.

RoboTurk Can Pick and Place - Crowdsourced Demonstrations: We leverage the RoboTurk pilot dataset [26] to train policies on the Robosuite Can Pick and Place task [9]. While the original dataset contained over 1100 demonstrations, we present results on a filtered version consisting of the fastest 225 trajectories. These demonstrations were collected across multiple humans and exhibit significant suboptimality and diversity in the solution approaches. For example, some people chose to grasp the can in an upright position by carefully positioning the gripper above the can while others chose to knock the can over before grasping the can on its side. An example of the latter is shown in Fig. 2. This dataset contains 225 demonstrations with an average completion time of 589 timesteps.

RoboTurk Can Image: This is a variant of the crowd-sourced dataset that has image observations from a frontview camera instead of robot and object observations.

B. Experiment Details

We compare IRIS to a Behavioral Cloning (BC) baseline that performs simple regression over state-action pairs in the dataset, a Recurrent Neural Network (RNN) variant of Behavioral Cloning that we call BC-RNN, and a Batch-Constrained Q-Learning (BCQ) baseline, which is a state-of-the-art Batch Reinforcement Learning algorithm for continuous control [11]. We also compare against two variants of IRIS to evaluate the utility of each component - a version with no Q-function at the high-level (goal selection occurs by simply sampling the Goal VAE) and a version where a deterministic goal prediction network is used in lieu of the VAE (simple regression is used to train this network). We emphasize that all training is *offline* - no algorithm is allowed to collect additional samples.

VII. Results

1. Can IRIS successfully recover a performant policy by *selectively imitating* pieces of a varied set of demonstrations? To answer this, we present quantitative results across all datasets and baselines in Table I and also investigate qualitative model performance on the Graph Reach dataset in Fig. 4. Table I shows that while all models are able to solve the Graph Reach task consistently, variants of IRIS and BCQ are able to solve the task faster. To verify that

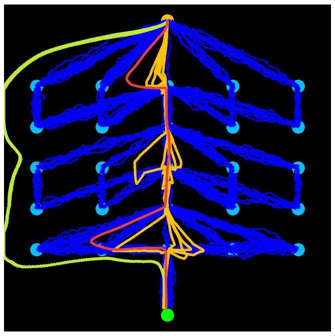

Fig. 4: **Qualitative Evaluation:** We visualize 5 trajectories taken by the best performing policies for the BC (red), BCQ (green), and IRIS (orange) models on the Graph Reach environment. A set of 50 trajectories from the dataset (blue) is also shown. Our model is both able to faithfully reconstruct demonstrated trajectories and leverage them to reach the goal quickly. By contrast, BCQ extrapolates an entirely new trajectory, while BC converges to a particular mode in the dataset that is slow to reach the goal. Unlike the other models, ours also exhibits variation in policy rollouts.

IRIS is indeed imitating useful portions of demonstrated trajectories, we plot trajectories taken by the best BC (red), BCQ (green) and IRIS (orange) model in Fig. 4 and compare them to trajectories in the dataset (shown in blue). The plot demonstrates that our model has the capacity to imitate several different demonstrated modes from the dataset and leverage them to reach the goal quickly, while BCQ extrapolates to unseen states to reach the goal and attain similar performance. This type of extrapolation can be harmful in more complex robot manipulation tasks such as our Lift and Can tasks. This experiment demonstrates that IRIS is able to reproduce multimodal behaviors in the dataset and selectively interpolate between them to solve a task efficiently.

2. What benefits do our two-level decomposition provide for learning manipulation policies from diverse demonstration data? We consider the more challenging Lift and Cans manipulation datasets. As Table I and the left two plots of Fig. 3 show, there is a stark contrast in performance between variants of IRIS and baselines. Our models achieve success rates of 70-80% and 20-30% on the Lift and Cans tasks respectively while baseline models can only attain 18% on the Lift task, and fail to solve the Cans task at all. The only difference between the BC-RNN model and IRIS, no Goal VAE is that IRIS conditions the RNN on goal observations and these goal observations are generated at test-time by a network that was trained to predict observations T timesteps into the future. The large performance gap between these two models implies that *goal-directed imitation*, which the baselines lack, is critical to deal with the multimodality in these datasets, and helps facilitate faithful imitation.

Allowing for diverse goal predictions also significantly improves performance - IRIS, no Q achieves 10% higher success rate than IRIS, no Goal VAE on the Cans dataset by replacing a deterministic goal prediction with a VAE. Finally, although using the value network for goal selection did not improve performance on the Cans dataset, using value selection allowed significant improvement on the Lift task. We hypothesize that the value function helps avoid situations where the demonstrator moved away from the cube or drifted from side to side on the Lift dataset by choosing goals that lead the arm closer to the cube. In summary, our decomposition allows behaviors from the demonstrations to be reproduced over an extended period of time while simultaneously allowing the high-level component flexibility in dictating which behaviors should be reproduced.

3. How much data is necessary to train policies successfully on these tasks? We train IRIS on smaller subsets of the datasets - small datasets consisting of the best 10% of the trajectories (in terms of completion time) and medium datasets consisting of the best 50% of the trajectories. The right two plots in Fig. 3 depict learning curves for IRIS on these datasets. The smaller-sized datasets lead to poor performance but the medium-sized Lift dataset has the same asymptotic performance as the full dataset. By contrast, the medium-sized Cans dataset restricts performance significantly. This shows that for tasks with greater variation in task instance, IRIS benefits from having more data in the dataset.

4. Can IRIS train successful policies on datasets with image observations? We train IRIS on the RoboTurk Cans Image dataset, where the observations are 128 by 128 RGB images of the robot workspace from a camera placed in front of the robot arm. We leverage the method from Dundar et al. [8] to pre-train a landmark representation for all image observations. We use 16 landmark locations, corresponding to a 32-dimensional representation for each image. Then, we train IRIS on these representations. Surprisingly, all three IRIS variants achieve higher success rates than their counterparts on the low dimensional dataset - with the best model achieving 42.7% success rate. This result suggests that IRIS can leverage pre-trained low-dimensional representations of high-dimensional observations in order to learn performant visuomotor policies from completely offline data.

VIII. Conclusion

We introduced IRIS, a framework for offline learning from a large set of diverse and suboptimal demonstrations that operates by selectively imitating local sequences from the dataset. We demonstrated that IRIS recovers performant policies from large manipulation datasets and significantly outperforms other baselines due to our decomposition of the problem into *goal-conditioned imitation* and a high-level goal selection mechanism. One limitation of the current approach is that training and testing distributions of task instances must be similar. For example, the training data must contain a sufficient variety of initial can locations to expect that the test-time policy can generalize to all can locations inside the bin. Domain adaptation for dealing with novel test-time scenarios is an exciting direction for future work.

Acknowledgment

Ajay Mandlekar acknowledges the support of the Department of Defense (DoD) through the NDSEG program. We thank Kevin Shih for help with training models on image observations. We thank members of the NVIDIA Seattle Robotics Lab for several helpful discussions and feedback.

References

[1] P. Abbeel and A. Y. Ng, "Inverse reinforcement learning", in *Encyclopedia of machine learning*, Springer, 2011, pp. 554–558.

[2] J. Achiam, E. Knight, and P. Abbeel, "Towards characterizing divergence in deep q-learning", *arXiv preprint arXiv:1903.08894*, 2019.

[3] R. Agarwal, D. Schuurmans, and M. Norouzi, "Striving for simplicity in off-policy deep reinforcement learning", *arXiv preprint arXiv:1907.04543*, 2019.

[4] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. P. Abbeel, and W. Zaremba, "Hindsight experience replay", in *Advances in Neural Information Processing Systems*, 2017, pp. 5048–5058.

[5] A. Bhatt, M. Argus, A. Amiranashvili, and T. Brox, "Crossnorm: Normalization for off-policy td reinforcement learning", *arXiv preprint arXiv:1902.05605*, 2019.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database", in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[7] Y. Ding, C. Florensa, M. Phielipp, and P. Abbeel, "Goal-conditioned imitation learning", *arXiv preprint arXiv:1906.05838*, 2019.

[8] A. Dundar, K. J. Shih, A. Garg, R. Pottorf, A. Tao, and B. Catanzaro, "Unsupervised disentanglement of pose, appearance and background from images and videos", *arXiv preprint arXiv:2001.09518*, 2020.

[9] L. Fan, Y. Zhu, J. Zhu, Z. Liu, O. Zeng, A. Gupta, J. Creus-Costa, S. Savarese, and L. Fei-Fei, "Surreal: Open-source reinforcement learning framework and robot manipulation benchmark", in *Conference on Robot Learning*, 2018.

[10] K. Fang, Y. Zhu, A. Garg, A. Kurenkov, V. Mehta, L. Fei-Fei, and S. Savarese, "Learning task-oriented grasping for tool manipulation from simulated self-supervision", in *Robotics: Systems and Science*, 2018.

[11] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration", *arXiv preprint arXiv:1812.02900*, 2018.

[12] Y. Gao, J. Lin, F. Yu, S. Levine, T. Darrell, *et al.*, "Reinforcement learning from imperfect demonstrations", *arXiv preprint arXiv:1802.05313*, 2018.

[13] C. Goldfeder, M. Ciocarlie, H. Dang, and P. K. Allen, "The columbia grasp database", in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, IEEE, 2009, pp. 1710–1716.

[14] T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, D. Horgan, J. Quan, A. Sendonaris, I. Osband, *et al.*, "Deep q-learning from demonstrations", in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[15] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "Beta-vae: Learning basic visual concepts with a constrained variational framework.", *ICLR*, vol. 2, no. 5, p. 6, 2017.

[16] N. Jaques, A. Ghandeharioun, J. H. Shen, C. Ferguson, A. Lapedriza, N. Jones, S. Gu, and R. Picard, "Way off-policy batch deep reinforcement learning of implicit human preferences in dialog", *arXiv preprint arXiv:1907.00456*, 2019.

[17] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, *et al.*, "Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation", *arXiv preprint arXiv:1806.10293*, 2018.

[18] A. Kasper, Z. Xue, and R. Dillmann, "The kit object models database: An object model database for object recognition, localization and manipulation in service robotics", *The International Journal of Robotics Research*, vol. 31, no. 8, pp. 927–934, 2012.

[19] D. P. Kingma and M. Welling, "Auto-encoding variational bayes", *arXiv preprint arXiv:1312.6114*, 2013.

[20] S. Krishnan, A. Garg, R. Liaw, B. Thananjeyan, L. Miller, F. T. Pokorny, and K. Goldberg, "Swirl: A sequential windowed inverse reinforcement learning algorithm for robot tasks with delayed rewards", *The International Journal of Robotics Research*, 2019.

[21] A. Kumar, J. Fu, G. Tucker, and S. Levine, "Stabilizing off-policy q-learning via bootstrapping error reduction", *arXiv preprint arXiv:1906.00949*, 2019.

[22] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, "Learning hand-eye coordination for robotic grasping with large-scale data collection", in *ISER*, 2016, pp. 173–184.

[23] Y. Liu, A. Swaminathan, A. Agarwal, and E. Brunskill, "Off-policy policy gradient with state distribution correction", *arXiv preprint arXiv:1904.08473*, 2019.

[24] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet, "Learning latent plans from play", *arXiv preprint arXiv:1903.01973*, 2019.

[25] 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.

[26] A. Mandlekar, Y. Zhu, A. Garg, J. Booher, M. Spero, A. Tung, J. Gao, J. Emmons, A. Gupta, E. Orbay, S. Savarese, and L. Fei-Fei, "RoboTurk: A Crowdsourcing Platform for Robotic Skill Learning through Imitation", in *Conference on Robot Learning*, 2018.

[27] O. Nachum, S. S. Gu, H. Lee, and S. Levine, "Data-efficient hierarchical reinforcement learning", in *Advances in Neural Information Processing Systems*, 2018, pp. 3303–3313.

[28] A. Nair, B. McGrew, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Overcoming exploration in reinforcement learning with demonstrations", in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2018, pp. 6292–6299.

[29] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours", in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, IEEE, 2016.

[30] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network", in *Advances in neural information processing systems*, 1989, pp. 305–313.

[31] V. Pong, S. Gu, M. Dalal, and S. Levine, "Temporal difference models: Model-free deep rl for model-based control", *arXiv preprint arXiv:1802.09081*, 2018.

[32] P. Rajpurkar, R. Jia, and P. Liang, "Know What You Don't Know: Unanswerable Questions for SQuAD", *arXiv preprint arXiv:1806.03822*, 2018.

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

[34] S. Ross, N. Melik-Barkhudarov, K. S. Shankar, A. Wendel, D. Dey, J. A. Bagnell, and M. Hebert, "Learning monocular reactive uav control in cluttered natural environments", in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, IEEE, 2013, pp. 1765–1772.

[35] J. Schulman, J. Ho, C. Lee, and P. Abbeel, "Learning from demonstrations through the use of non-rigid registration", in *Robotics Research*, Springer, 2016, pp. 339–354.

[36] M. Večerík, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe, and M. Riedmiller, "Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards", *arXiv preprint arXiv:1707.08817*, 2017.

[37] K.-T. Yu, M. Bauza, N. Fazeli, and A. Rodriguez, "More than a million ways to be pushed. a high-fidelity experimental dataset of planar pushing", in *Int'l Conference on Intelligent Robots and Systems*, 2016.

[38] Y. Zhu, Z. Wang, J. Merel, A. Rusu, T. Erez, S. Cabi, S. Tunyasuvunakool, J. Kramár, R. Hadsell, N. de Freitas, *et al.*, "Reinforcement and imitation learning for diverse visuomotor skills", in *RSS*, 2018.