arXiv:1802.01557v1 [cs.LG] 5 Feb 2018

One-Shot Imitation from Observing Humans via Domain-Adaptive Meta-Learning

Tianhe Yu*, Chelsea Finn*, Annie Xie, Sudeep Dasari, Tianhao Zhang, Pieter Abbeel, Sergey Levine
University of California, Berkeley
Email: {tianhe.yu,cbfinn,anniexie,sdasari,tianhao.z,pabbeel,svlevine}@berkeley.edu
* denotes equal contribution

***Abstract*—Humans and animals are capable of learning a new behavior by observing others perform the skill just once. We consider the problem of allowing a robot to do the same – learning from a raw video pixels of a human, even when there is substantial domain shift in the perspective, environment, and embodiment between the robot and the observed human. Prior approaches to this problem have hand-specified how human and robot actions correspond and often relied on explicit human pose detection systems. In this work, we present an approach for one-shot learning from a video of a human by using human and robot demonstration data from a variety of previous tasks to build up prior knowledge through meta-learning. Then, combining this prior knowledge and only a single video demonstration from a human, the robot can perform the task that the human demonstrated. We show experiments on both a PR2 arm and a Sawyer arm, demonstrating that after meta-learning, the robot can learn to place, push, and pick-and-place new objects using just one video of a human performing the manipulation.**

I. Introduction

Demonstrations provide a descriptive medium for specifying robotic tasks. Prior work has shown that robots can acquire a range of complex skills through demonstration, such as table tennis [28], lane following [34], pouring water [31], drawer opening [38], and multi-stage manipulation tasks [62]. However, the most effective methods for robot imitation differ significantly from how humans and animals might imitate behaviors: while robots typically need to receive demonstrations in the form of kinesthetic teaching [32, 1] or teleoperation [8, 35, 62], humans and animals can acquire the gist of a behavior simply by *watching* someone else. In fact, we can adapt to variations in morphology, context, and task details effortlessly, compensating for whatever *domain shift* may be present and recovering a skill that we can use in new situations [6]. Additionally, we can do this from a very small number of demonstrations, often only one. How can we endow robots with the same ability to learn behaviors from raw third person observations of human demonstrators?

Acquiring skills from raw camera observations presents two major challenges. First, the difference in appearance and morphology of the human demonstrator from the robot introduces a systematic domain shift, namely the correspondence problem [29, 6]. Second, learning from raw visual inputs typically requires a substantial amount of data, with modern deep learning vision systems using hundreds of thousands to millions of images [57, 18]. In this paper, we demonstrate that we can begin to address both of these challenges through a single approach based on meta-learning. Instead of manually specifying the correspondence between human and robot, which can be particularly complex for skills where different morphologies require different strategies, we propose a data-driven approach. Our approach can acquire new skills from only one video of a human. To enable this, it builds a rich prior over tasks during a *meta-training* phase, where both human demonstrations and teleoperated demonstrations are available for a variety of other, structurally similar tasks. In essence, the robot learns how to learn from humans using this data. After the meta-training phase, the robot can acquire new skills by combining its learned prior knowledge with one video of a human performing the new skill.

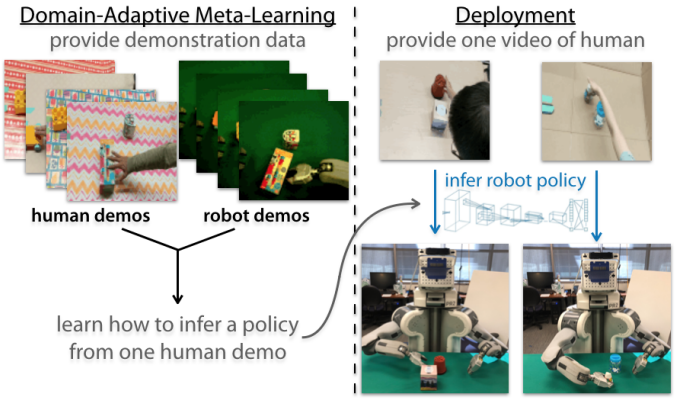

Fig. 1. After meta-learning with human and robot demonstration data, the robot learns to recognize and push a new object from one video of a human.

The main contribution of this paper is a system for learning robotic manipulation skills from a single video of a human by leveraging large amounts of prior meta-training data, collected for different tasks. When deployed, the robot can adapt to a particular task with novel objects using just a single video of a human performing the task with those objects (e.g., see Figure 1). The video of the human need not be from the same perspective as the robot, or even be in the same room. The robot is trained using videos of humans performing tasks with various objects along with demonstrations of the robot performing the same task. Our experiments on two real robotic platforms demonstrate the ability to learn directly from RGB videos of humans, and to handle novel objects, novel humans, and videos of humans in novel scenes. Videos of the results can be found on the supplementary website.[1]

[1] The video is available at https://sites.google.com/view/daml

II. Related Work

Most imitation learning and learning from demonstration methods operate at the level of configuration-space trajectories [44, 2], which are typically collected using kinesthetic teaching [32, 1], teleoperation [8, 35, 62], or sensors on the demonstrator [11, 9, 7, 21]. Instead, can we allow robots to imitate just by watching the demonstrator perform the task? We focus on this problem of learning from one video demonstration of a human performing a task, in combination with human and robot demonstration data collected on other tasks. Prior work has proposed to resolve the correspondence problem by hand, for example, by manually specifying how human grasp poses correspond to robot grasps [20] or by manually defining how human activities or commands translate into robot actions [58, 23, 30, 37, 40]. By utilizing demonstration data of how humans and robots perform each task, our approach learns the correspondence between the human and robot implicitly. Many prior approaches also use explicit hand-tracking systems with carefully engineered pipelines for visual activity recognition [22, 37]. In contrast to such approaches, which rely on precise hand detection and a pre-built vision system, our approach is trained end-to-end, seeking to extract the aspects of the human's activity that are the most relevant for choosing actions. This places less demand on the vision system, requiring it only to implicitly deduce the task and how to accomplish it, rather than precisely tracking the human's body and nearby objects.

Other prior approaches have sought to solve the problem of learning from human demonstrations by explicitly determining the goal or reward underlying the human behavior (e.g. through inverse reinforcement learning), and then optimizing the reward through reinforcement learning (RL). For example, Rhinehart and Kitani [39] and Tow et al. [53] learn a model that predicts the outcome of the human's demonstration from a particular scene. Similarly, other works have learned a reward function based on human demonstrations [47, 49, 26, 46, 50]. Once the system has learned about the reward function or desired outcome underlying the given task, the robot runs some form of reinforcement learning to maximize the reward or to reach the desired outcome. This optimization typically requires substantial experience to be collected using the robot for each individual task. Other approaches assume a known model and perform trajectory optimization to reach the inferred goal [27]. Because all of these methods consider single tasks in isolation, they often require multiple human demonstrations of the new task (though not all, e.g. [46, 27]). Our method only requires one demonstration of the new task setting and, at test time, does not require additional experience on the robot nor a known model. And, crucially, all of the data used in our approach is amortized across tasks, such that the amount of data needed for any given individual task as quite small. In contrast, these prior reward-learning methods only handle the single-task setting, where a considerable amount of data must be collected for an individual task.

Instead of using the previously mentioned approaches, we use a meta-learning approach [51, 45], which in recent works has shown the ability to learn from just one demonstration [15, 10] using prior knowledge built up from many demonstrations of other tasks. In particular, we extend the approach of model-agnostic meta-learning [14, 15]. Up until now, these approaches have not considered the problem of domain shift between the demonstration used for training and for testing, e.g. learning from videos of humans.

Handling domain shift is a key aspect of this problem, with a shift in both the visual scene and the embodiment of the human/robot, including the degrees of freedom and the physics. Domain adaptation has received significant interest within the machine learning community, especially for varying visual domains [3, 33] and visual shift between simulation and reality [56, 41]. Many of these techniques aim to find a representation that is invariant to the domain [12, 16, 54, 41, 25]. Other approaches have sought to map datapoints from one domain to another [48, 59, 5, 60]. The human imitation problem may involve developing invariances, for example, to the background or lighting conditions of the human and robot's environments. However, the physical correspondence between human and robot does not call for an invariant representation, nor a direct mapping between the domains. In many scenarios, a direct physical correspondence between robot and human poses might not exist. Instead, the system must implicitly recognize the goal of the human from the video and determine the appropriate action.

III. Preliminaries

Our approach builds upon prior work in learning to learn or meta-learning, in order to learn how to infer a robot policy from just one human demonstration. In particular, we will present an extension of the model-agnostic meta-learning algorithm (MAML) [14] that allows for the ability to handle domain shift between the provided data (i.e. a human demo) and the evaluation setting (i.e. the robot's execution), and the ability to learn effectively without labels (i.e., the human's actions). In this section, we will overview the general meta-learning problem and the MAML algorithm.

Meta-learning algorithms optimize for the ability to learn new tasks quickly and efficiently. To do so, they use data collected across a wide range of meta-training tasks and are evaluated based on their ability to learn new meta-test tasks. Meta-learning assumes that the meta-training and meta-test tasks are drawn from some distribution $p(\mathcal{T})$. Generally, meta-learning can be viewed as discovering the structure that exists between tasks such that, when the model is presented with a new task from the meta-test set, it can use the known structure to quickly learn the task. MAML achieves this by optimizing for a deep network's initial parameter setting such that one or a few steps of gradient descent on a few datapoints leads to effective generalization (referred to as few-shot generalization) [14]. Then, after meta-training, the learned parameters are fine-tuned on data from a new task. This meta-learning process can be formalized as learning a prior over functions, and the fine-tuning process as inference under the learned prior [42, 17].

Concretely, consider a supervised learning problem with a loss function denoted as $\mathcal{L}(\theta, \mathcal{D})$, where θ denotes the model parameters and $\mathcal{D}$ denotes the labeled data. For a few-shot supervised learning problem, MAML assumes access to a small amount of data for a large number of tasks. During meta-training, a task $\mathcal{T}$ is sampled, along with data from that task, which is randomly partitioned into two sets, $\mathcal{D}^{\text{tr}}$ and $\mathcal{D}^{\text{val}}$. We will assume that $\mathcal{D}^{\text{tr}}$ has K examples. MAML optimizes for a set of model parameters θ such that one or a few gradient steps on $\mathcal{D}^{\text{tr}}$ produces good performance on $\mathcal{D}^{\text{val}}$. Effectively, MAML optimizes for generalization from K examples. Thus, using $\phi_{\mathcal{T}}$ to denote the updated parameters, the MAML objective is the following:

$$\min_{\theta} \sum_{\mathcal{T}} \mathcal{L}(\theta - \alpha \nabla_{\theta} \mathcal{L}(\theta, \mathcal{D}^{\text{tr}}_{\mathcal{T}}), \mathcal{D}^{\text{val}}_{\mathcal{T}}) = \min_{\theta} \sum_{\mathcal{T}} \mathcal{L}(\phi_{\mathcal{T}}, \mathcal{D}^{\text{val}}_{\mathcal{T}}).$$

where α is a step size that can be set as a hyperparameter or learned. Moving forward, we will refer to the inner loss function as the *adaptation objective* and the outer objective as the *meta-objective*. Subsequently, at meta-test time, K examples from a new, held-out task $\mathcal{T}_{\text{test}}$ are presented and we can run gradient descent starting from θ to infer model parameters for the new task:

$$\phi_{\mathcal{T}_{\text{test}}} = \theta - \alpha \nabla_{\theta} \mathcal{L}(\theta, \mathcal{D}^{\text{tr}}_{\mathcal{T}_{\text{test}}}).$$

For convenience, we will use only one inner gradient step in the equations. However, using multiple inner gradient steps is straight-forward, and frequently done in practice.

Finn et al. [15] applied the MAML algorithm to one-shot imitation learning problem, using robot demonstrations collected via teleoperation and a mean-squared error behavioral cloning objective for the loss $\mathcal{L}$. While this enables learning from one robot demonstration at meta-test time, it does not allow the robot to learn from a raw video of a human or handle domain shift between the demonstration medium and the robot. Next, we will present our approach for one-shot imitation learning from raw video under domain shift.

IV. Learning from Humans

In this section, we will present the problem statement of one-shot imitation learning from humans, introduce our method, and discuss a key aspect of our approach: a learned temporal adaptation objective.

A. Problem Overview

The problem of learning from human video can be viewed as an inference problem, where the goal is to infer the robot policy parameters $\phi_{\mathcal{T}_i}$ that will accomplish the task $\mathcal{T}_i$ by incorporating prior knowledge with a small amount of evidence, in the form of one human demonstration. In order to effectively learn from just one video of a human, we need a rich prior that encapsulates a visual and physical understanding of the world, what kinds of outcomes the human might want to accomplish, and which actions might allow a robot to bring about that outcome. We could choose to encode prior knowledge manually, for example by using a pre-defined vision system, a pre-determined set of human objectives, or a known dynamics model. However, this type of manual knowledge encoding is task-specific and time-consuming, and does not benefit from data. We will instead study how we can learn this prior automatically, using human and robot demonstration data from a variety of tasks.

Formally, we will define a demonstration from a human $\mathbf{d}^h$ to be a sequence of image observations $\mathbf{o}_1, ..., \mathbf{o}_T$, and a robot demonstration $\mathbf{d}^r$ to be a sequence of image observations, robot states, and robot actions: $\mathbf{o}_1, \mathbf{s}_1, \mathbf{a}_1, ..., \mathbf{o}_T, \mathbf{s}_T, \mathbf{a}_T$. The robot state includes the robot's body configuration, such as joint angles, but does not include object information, which must be inferred from the image. We do not make any assumptions about the similarities or differences between the human and robot observations; they may contain substantial domain shift, e.g. differences in the appearance of the arms, background clutter, and camera viewpoint.

Our approach consists of two phases. First, in the meta-training phase, the goal will be to acquire a prior over policies using both human and robot demonstration data, that can then be used to quickly learn to imitate new tasks with only human demonstrations. For meta-training, we will assume a distribution over tasks $p(\mathcal{T})$, a set of tasks $\{\mathcal{T}_i\}$ drawn from $p(\mathcal{T})$ and, for each task, two small datasets containing several human and robot demonstrations, respectively: $(\mathcal{D}^h_{\mathcal{T}_i}, \mathcal{D}^r_{\mathcal{T}_i})$. After the meta-training phase, the learned prior can be used in the second phase, when the method is provided with a human demonstration of a new task $\mathcal{T}$ drawn from $p(\mathcal{T})$. The robot must combine its prior with the new human demonstration to infer policy parameters $\phi_{\mathcal{T}}$ that solve the new task. We will next discuss our approach in detail.

B. Domain-Adaptive Meta-Learning

We develop a domain-adaptive meta-learning method, which will allow us to handle the setting of learning from video demonstrations of humans. While we will extend the MAML algorithm for this purpose, the key idea of our approach is applicable to other meta-learning algorithms. Like the MAML algorithm, we will learn a set of initial parameters, such that after one or a few steps of gradient descent on just one human demonstration, the model can effectively perform the new task. Thus, the data $\mathcal{D}^{\text{tr}}_{\mathcal{T}}$ will contain one human demonstration of task $\mathcal{T}$, and the data $\mathcal{D}^{\text{val}}_{\mathcal{T}}$ will contain one or more robot demonstrations of the same task.

Unfortunately, we cannot use a standard imitation learning loss for the inner adaptation objective computed using $\mathcal{D}^{\text{tr}}_{\mathcal{T}}$, since we do not have access to the human's actions. Even if we knew the human's actions, they will typically not correspond directly to the robot's actions. Instead, we propose to meta-learn an adaptation objective that does not require actions, and instead operates only on the policy activations. The intuition behind meta-learning a loss function is that we can acquire a function that only needs to look at the available inputs (which do not include the actions), and still produce gradients that are suitable for updating the policy parameters so that it can produce effective actions after the gradient update. While this might seem like an impossible task, it is important to

Algorithm 1 Meta-imitation learning from humans

Require: $\{(\mathcal{D}^h_{\mathcal{T}_i}, \mathcal{D}^r_{\mathcal{T}_i})\}$: human and robot demonstration data for a set of tasks $\{\mathcal{T}_i\}$ drawn from $p(\mathcal{T})$
Require: α, β: inner and outer step size hyperparameters
  while training **do**
    Sample task $\mathcal{T} \sim p(\mathcal{T})$ {or minibatch of tasks}
    Sample video of human $\mathbf{d}^h \sim \mathcal{D}^h_{\mathcal{T}}$
    Compute policy parameters $\phi_{\mathcal{T}} = \theta - \alpha \nabla_\theta \mathcal{L}_\psi(\theta, \mathbf{d}^h)$
    Sample robot demo $\mathbf{d}^r \sim \mathcal{D}^r_{\mathcal{T}}$
    Update $(\theta, \psi) \leftarrow (\theta, \psi) - \beta \nabla_{\theta,\psi} \mathcal{L}_{\text{BC}}(\phi_{\mathcal{T}}, \mathbf{d}^r)$
  end while
  Return θ, ψ

remember that the meta-training process still supervises the policy with true robot actions during meta-training. The role of the adaptation loss therefore may be interpreted as simply directing the policy parameter update to modify the policy to pick up on the right visual cues in the scene, so that the meta-trained action output will produce the right actions. We will discuss the particular form of $\mathcal{L}_\psi$ in the following section.

During the meta-training phase, we will learn both an initialization θ and the parameters ψ of the adaptation objective $\mathcal{L}_\psi$. The parameters θ and ψ are optimized for choosing actions that match the robot demonstrations in $\mathcal{D}^{\text{val}}_{\mathcal{T}}$. After meta-training, the parameters θ and ψ are retained, while the data is discarded. A human demonstration $\mathbf{d}^h$ is provided for a new task $\mathcal{T}$ (but not a robot demonstration). To infer the policy parameters for the new task, we use gradient descent starting from θ using the learned loss $\mathcal{L}_\psi$ and one human demonstration $\mathbf{d}^h$: $\phi_{\mathcal{T}} = \theta - \alpha \nabla_\theta \mathcal{L}_\psi(\theta, \mathbf{d}^h)$.

We optimize for task performance during meta-training using a behavioral cloning objective that maximizes the probability of the expert actions in $\mathcal{D}^{\text{val}}$. In particular, for a policy parameterized by ϕ that outputs a distribution over actions $\pi_\phi(\cdot|\mathbf{o}, \mathbf{s})$, the behavioral cloning objective is

$$\mathcal{L}_{\text{BC}}(\phi, \mathbf{d}^r) = \mathcal{L}_{\text{BC}}(\phi, \{\mathbf{o}_{1:T}, \mathbf{s}_{1:T}, \mathbf{a}_{1:T}\}) = \sum_t \log \pi_\phi(\mathbf{a}_t|\mathbf{o}_t, \mathbf{s}_t)$$

Putting this together with the inner gradient descent adaptation, the meta-training objective is the following:

$$\min_{\theta,\psi} \sum_{\mathcal{T}\sim p(\mathcal{T})} \sum_{\mathbf{d}^h \in \mathcal{D}^h_{\mathcal{T}}} \sum_{\mathbf{d}^r \in \mathcal{D}^r_{\mathcal{T}}} \mathcal{L}_{\text{BC}}(\theta - \alpha \nabla_\theta \mathcal{L}_\psi(\theta, \mathbf{d}^h), \mathbf{d}^r).$$

The algorithm for optimizing this meta-objective is summarized in Algorithm 1, whereas the procedure for learning from humans at meta-test time is shown in Algorithm 2. We will next discuss the form of the learned loss function, $\mathcal{L}_\psi$, which is critical for effective learning.

C. Learned Temporal Adaptation Objectives

To learn from a video of a human, we need an adaptation objective that can effectively capture relevant information in the video, such as the intention of the human and the task-relevant objects. While a standard behavior cloning loss is applied to each time step independently, the learned adaptation objective

Algorithm 2 Learning from human video after meta-learning

Require: meta-learned initial policy parameters θ
Require: learned adaptation objective $\mathcal{L}_\psi$
Require: one video of human demo $\mathbf{d}^h$ for new task $\mathcal{T}$
  Compute policy parameters $\phi_{\mathcal{T}} = \theta - \alpha \nabla_\theta \mathcal{L}_\psi(\theta, \mathbf{d}^h)$
  return π_ϕ

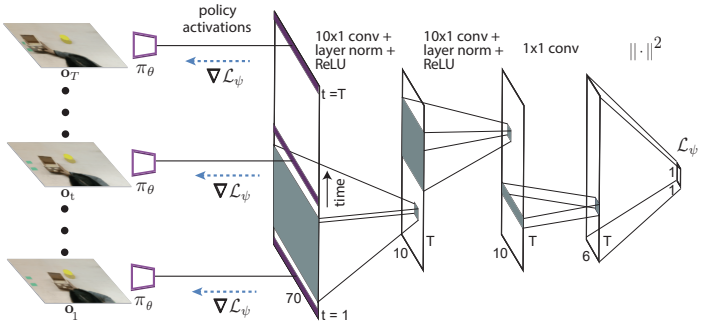

Fig. 2. Visualization of the learned adaptation objective, which uses temporal convolutions to integrate temporal information in the video demonstration.

must solve a harder task, since it must provide the policy with suitable gradient information *without* access to the true actions. As discussed previously, this is still possible, since the policy is trained to output good actions during meta-training. The learned loss must simply supply the gradients that are needed to modify the perceptual components of the policy to attend to the right objects in the scene, so that the action output actually performs the right task. However, determining which behavior is being demonstrated and which objects are relevant will often require examining multiple frames at the same time to determine the human's motion. To incorporate this temporal information, our learned adaptation objective therefore couples multiple time steps together, operating on policy activations from multiple time steps.

Since temporal convolutions have been shown to be effective at processing temporal and sequential data [55], we choose to adopt a convolutional network to represent the adaptation objective $\mathcal{L}_\psi$, using multiple layers of 1D convolutions over time. We choose to use temporal convolutions over a more traditional recurrent neural network like an LSTM, since they are simpler and usually more parameter efficient [55]. See Figure 2 for a visualization.

Prior work introduced a two-head architecture for one-shot imitation, with one head used for the pre-update demonstration and one head used for the post-update policy [15]. The two-head architecture can be interpreted as a learned linear loss function operating on the last hidden layer of the policy network for a particular timestep. The loss and the gradient are then computed by averaging over all timesteps in the demonstration. As discussed previously, a single timestep of an observed video is often not sufficient for learning from video demonstrations without actions. Thus, this simple averaging scheme is not effective at integrating temporal information. In Section VI, we show that our learned temporal loss can enable effective learning from demonstrations without actions,

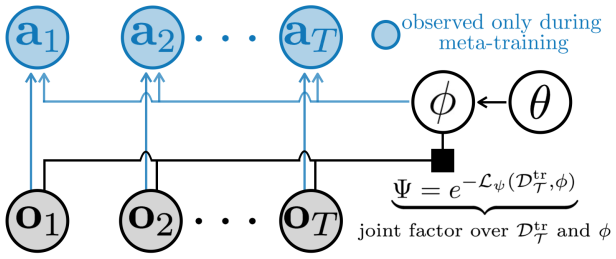

Fig. 3. Graphical model underlying our approach. During meta-training, both the observations $\mathbf{o}_t$ and the actions $\mathbf{a}_t$ are observed, and our method learns θ and Ψ. During meta-testing, only the observations are available, from which our method combines with the learned prior θ and factor Ψ to infer the task-specific policy parameters ϕ.

substantially outperforming this single-timestep linear loss.

D. Probabilistic Interpretation

One way to interpret meta-learning with learned adaptation objectives is by casting it into the framework of probabilistic graphical models. We can accomplish this by building on a derivation proposed in prior work [17], which frames MAML as approximately inferring a posterior over policy parameters ϕ given the evidence $\mathcal{D}^{\text{tr}}_{\mathcal{T}} = \mathbf{d}^h_{\mathcal{T}}$ (the data for a new task $\mathcal{T}$) and a prior over the parameters, given by θ. This prior work shows that a few steps of gradient descent on the likelihood $\log p(\mathcal{D}^{\text{tr}}_{\mathcal{T}}|\phi)$ starting from $\phi = \theta$ are approximately equivalent to maximum a posteriori (MAP) inference on $\log p(\phi|\mathcal{D}^{\text{tr}}_{\mathcal{T}}, \theta)$, where θ induces a Gaussian prior on the weights with mean θ and a covariance that depends on the step size and number of gradient steps.[2] The derivation is outside of the scope of this paper, and we refer the reader to prior work for details [17, 43].

In our approach, adaptation involves gradient descent on the learned loss $\mathcal{L}_\psi(\phi, \mathcal{D}^{\text{tr}}_{\mathcal{T}})$, rather than the likelihood $\log p(\mathcal{D}^{\text{tr}}_{\mathcal{T}}|\phi)$. Since we still take a fixed number of steps of gradient descent starting from θ, the result in prior work still implies that we are approximately imposing the Gaussian prior $\log p(\phi|\theta)$ [17, 43], and therefore are performing approximate MAP inference on the following joint distribution:

$$p(\phi|\mathcal{D}^{\text{tr}}_{\mathcal{T}}, \theta) \propto p(\phi, \mathcal{D}^{\text{tr}}_{\mathcal{T}}|\theta) \propto \underbrace{p(\phi|\theta)}_{\text{from GD}} \quad \underbrace{\Psi(\phi, \mathcal{D}^{\text{tr}}_{\mathcal{T}})}_{\exp(-\mathcal{L}_\psi(\phi, \mathcal{D}^{\text{tr}}_{\mathcal{T}}))} .$$

This is a partially directed factor graph with a learned factor Ψ over ϕ and $\mathcal{D}^{\text{tr}}_{\mathcal{T}}$ that has the log-energy $\mathcal{L}_\psi(\phi, \mathcal{D}^{\text{tr}}_{\mathcal{T}})$. Bayesian inference would require integrating out ϕ, but MAP inference provides a tractable alternative that still produces good results in practice [17]. Training is performed by directly maximizing $\mathcal{L}_{\text{BC}}(\phi_{\mathcal{T}}, \mathcal{D}^{\text{tr}}_{\mathcal{T}})$, where $\phi_{\mathcal{T}}$ is the MAP estimate of ϕ. Since the behavior cloning loss corresponds to the log likelihood of the actions under a Gaussian mixture policy, we directly train both the prior θ and the log-energy $\mathcal{L}_\psi$ such that MAP inference maximizes the log probability of the actions. Note that, since we use MAP inference during training, the model does not necessarily provide well-calibrated probabilities. However, the probabilistic interpretation still helps to shed light on the role of the learned adaptation objective $\mathcal{L}_\psi$, which is to induce a joint factor on the observations in $\mathcal{D}_{\mathcal{T}}$ and the policy parameters ϕ. A visual illustration of the corresponding graphical model is shown in Figure 3.

[2]This result is exact in the case of linear functions, and a local approximation in the case of nonlinear neural networks.

V. Network Architectures

Now that we have presented our approach, we describe form of the policy π and the learned adaptation objective $\mathcal{L}_\psi$.

A. Policy Architecture

As illustrated in Figure 4, the policy architecture is a convolutional neural network that maps from RGB images to a distribution over actions. The convolutional network begins with a few convolutional layers, which are fed into a channel-wise spatial soft-argmax that extracts 2D feature points $\mathbf{f}$ for each channel of the last convolution layer [24]. Prior work has shown that the spatial soft-argmax is particularly effective and parameter-efficient for learning to represent the positions of objects in robotics domains [24, 13]. Following prior work [24], we concatenate these feature points with the robot configuration, which consists of the pose of the end-effector represented by the 3D position of 3 non-axis-aligned points on the gripper. Then, we pass the concatenated feature points and robot pose into multiple fully connected layers. The distribution over actions is predicted linearly from the last hidden layer $\mathbf{h}$. We initialize the first convolutional layer from that of a network trained on ImageNet.

In our experiments, we will be using a continuous action space over the linear and angular velocity of the robot's gripper and a discrete action space over the gripper open/close action. Gaussian mixtures can better model multi-modal action distributions compared to Gaussian distributions and has been used in previous imitation learning works [36]. Thus, for the continuous actions, we use a mixture density network [4] to represent the output distribution over actions. For the discrete action of opening or closing the gripper, we use a sigmoid output with a cross-entropy loss.

Following prior work [62], we additionally have the model predict the pose of the gripper when it contacts the target object and/or container. This is part of the outer meta-objective, and we can easily provide supervision using the robot demonstration. Note that this supervision is not needed at meta-test time when the robot is learning from a video of a human. For placing and pick-and-place tasks, the target container is located at the final end-effector pose. Thus, we use the last end-effector pose as supervision. For pushing and pick-and-place, the demonstrator manually specifies the time at which the gripper initially contacts the object and the end-effector pose at that time step is used. The model predicts this intermediate gripper pose linearly from the feature points $\mathbf{f}$, and the predicted pose is fed back into the policy. Further architecture details are included in Section VI.

B. Learned Adaptation Objective Architecture

Because we may need to update both the policy's perception and control, the adaptation objective will operate on a concatenation of the predicted feature points, $\mathbf{f}$ (at the end of

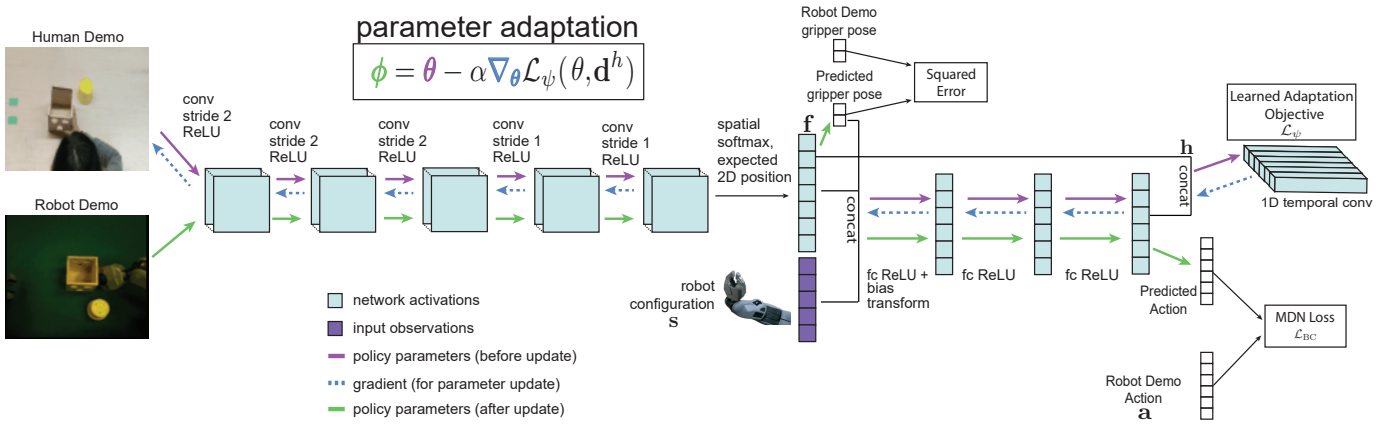

Fig. 4. Illustration of the policy architecture. The policy consists of a sequence of five convolutional (conv) layers, followed by a spatial soft-argmax and fully-connected (fc) layers. The learned adaptation objective $\mathcal{L}_\psi$ is further illustrated in Figure 2. Best viewed in color.

the perception layers), and the final hidden layer of the policy, $\mathbf{h}$ (at the end of the control layers). This allows the learned loss to more directly adapt the weights in the convolutional layers, bypassing the control layers. The updated task parameters are computed using our temporal adaptation objective,

$$\phi = \theta - \alpha \nabla_\theta \mathcal{L}_\psi(\theta, \mathbf{d}^h),$$

where we will decompose the objective into two parts: $\mathcal{L}_\psi = \mathcal{L}_{\psi_1}(\mathbf{f}_{1:T}) + \mathcal{L}_{\psi_2}(\mathbf{h}_{1:T})$ We use the same architecture for $\mathcal{L}_{\psi_1}$ and $\mathcal{L}_{\psi_2}$, which is illustrated in Figure 2. The learned objective consists of three layers of temporal convolutions, the first two with 10×1 filters and the third with 1×1 filters. The ℓ_2 norm of the output of the convolutions is computed to produce the scalar objective value.

VI. Experiments

Through our experiments, we aim to address three main questions: (1) Can our approach effectively learn a prior that enables the robot to learn to manipulate new objects after seeing just one video of a human? (2) Can our approach generalize to human demonstrations from a different perspective than the robot, on novel backgrounds, and with new human demonstrators? (3) How does the proposed approach compare to alternative approaches to meta-learning? In order to further understand our method and its applicability, we additionally evaluate it understand: (a) How important is the temporal adaptation objective? (b) Can our approach be used on more than one robot platform, and with either kinesthetic or teleoperated demonstrations for meta-training?

To answer these questions, we run our experiments primarily with a 7-DoF PR2 arm, with robot demonstrations collected via teleoperation, and RGB images collected from a consumer-grade camera (unless noted otherwise), and use a Sawyer robot with kinesthetic demonstrations to study (b). We compare the following meta-learning approaches:

- **contextual policy**: a feedforward network that takes as input the robot's observation and the final image of the human demo (to indicate the task), and outputs the predicted action.
- **DA-LSTM policy**: a recurrent network that directly ingests the human demonstration video and the current robot observation, and outputs the predicted robot action. This is a domain-adaptive version of the meta-learning algorithm proposed by Duan et al. [10].
- **DAML, linear loss**: our approach with a linear per-timestep adaptation objective.
- **DAML, temporal loss**: our approach with the temporal adaptation objective described in Section IV.

All methods use a mixture density network [4] to represent the action space, where the actions correspond to the linear and rotational velocity of the robot gripper, a 6-dimensional continuous action space. As discussed in Section V, each network is also trained to predict the final end-effector pose using a mean-squared error objective. We train each policy using the Adam optimizer with default hyperparameters [19]. All methods use the same data and receive the same supervision. For measuring generalization from meta-training to meta-testing, we use held-out objects in all of our evaluations that were not seeing during meta-training, as illustrated in Figure 7, and new human demonstrators. We provide full experimental details, hyperparameters, and architecture information in Appendix A. Code for our method will be released upon publication, and we encourage the reader to view the supplementary video.[3]

A. PR2 Placing, Pushing, and Pick & Place

We first consider three different task settings: placing a held object into a container while avoiding two distractor containers, pushing an object amid one distractor, and picking an object and placing it into a target container amid two distractor containers. The tasks are illustrated in Figure 5. In this initial experiment, we collect human demonstrations from the perspective of the robot's camera. For placing and pushing, we only use RGB images, whereas for pick-and-place, RGB-D is used. For meta-training, we collected a dataset with hundreds of objects, consisting of 1293, 640, and 1008 robot demonstrations for placing, pushing, and pick-and-place respectively, and an equal number of human demonstrations. We use the following metrics to define success for each task: for placing and pick & place, success if the object landed in or on any part of the correct container; for pushing, success if the correct item was pushed past or within $\sim$5 cm of the robot's left gripper.

[3]The video is available at sites.google.com/view/daml

Fig. 5. Example placing (left), pushing (middle), and pick-and-place (right) tasks, from the robot's perspective. The top row shows the human demonstrations used in Section VI-A, while the bottom shows the robot demonstration.

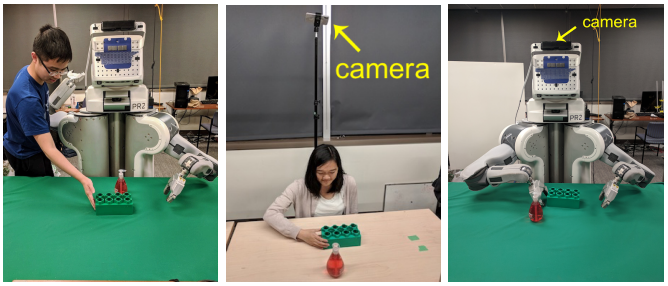

Fig. 6. The PR2 experimental set-up. Left & Middle: human demonstration set-up from Sections VI-A and VI-B respectively. Right: test-time set-up.

Fig. 7. Subset of the objects used for training and evaluation. The robot must learn to recognize and maneuver the novel test objects using just one video of a human.

During evaluation, we used 15, 12, and 15 novel target objects for placing, pushing, and pick-and-place respectively, collected one human demonstration per object, and evaluated three trials of the policy inferred from the human demonstration. We report the results in Table I. Our results show that, across the board, the robot is able to learn to interact with the novel objects using just one video of a human demo with that object, with pick-and-place being the most difficult task. We find that the DA-LSTM and contextual policies struggle, likely because they require more data to effectively infer the task. This finding is consistent with previous work [15]. Our results also indicate the importance of integrating temporal information when observing the human demonstration, as the linear loss performs poorly compared to using a temporal adaptation objective.

	placing	pushing	pick and place
DA-LSTM	33.3%	33.3%	5.6%
contextual	36.1%	16.7%	16.7%
DAML, linear loss	76.7%	27.8%	11.1%
DAML, temporal loss (ours)	**93.8%**	**88.9%**	**80.0%**

TABLE I. One-shot success rate of PR2 robot placing, pushing, and pick-and-place, using human demonstrations from the perspective of the robot. Evaluated using held-out objects and a novel human demonstrator.

B. Demonstrations with Large Domain Shift

Now, we consider a challenging setting, where human demonstrations are collected in a different room with a different camera and camera perspective from that of the robot. As a result, the background and lighting vary substantially from the robot's environment. We use a mounted cell-phone camera to record sequences of RGB images on ten different table textures, as illustrated in Figure 6. The corresponding view of the demonstrations is shown in Figure 8. We consider the pushing task, as described in Section VI-A, reusing the same robot demonstrations and collecting an equal number of new demonstrations. We evaluate performance on novel objects, a new human demonstrator, and with one seen and two novel backgrounds, as shown in Figure 9.

Like the previous experiment, we evaluated with 12 novel objects and 3 trials per object. Because we used different object pairs from the previous pushing experiment, the performance is not directly comparable to the results in Table I. The results for this experiment are summarized in Table II. As seen in the supplementary video, we find that the robot is able to successfully learn from the demonstrations with a different viewpoint and background. Performance degrades when using a novel background, which causes a varied shift in domain, but the robot is still able to perform the task about 70% of the time. In Table II, we also include an analysis of the failure modes of our approach, including the number of failures caused by incorrect task identification – misidentifying the object – versus control failures – when the object was clearly correctly identified, but the robot failed to effectively push it. We see that, when the human demonstrations are on a previously seen background, the robot only fails to identify the object once out of 33 trials, whereas failures of this kind are more frequent on the novel backgrounds. Collecting data on a more diverse array of backgrounds, or using some form of background augmentation would likely reduce these types of failures. The number of control failures is similar for all backgrounds, likely indicative of the challenge of physically maneuvering a variety of previously unseen objects.

pushing	seen bg	novel bg 1	novel bg 2
DAML, temporal loss (ours)	**81.8%**	**66.7%**	**72.7%**

Failure analysis of DAML	seen bg	novel bg 1	novel bg 2
# successes	27	22	24
# failures from task identification	1	5	4
# failures from control	5	6	5

TABLE II. Top: One-shot success rate of PR2 robot pushing, using videos of human demonstrations in a different scene and camera, with seen and novel backgrounds. Evaluated using held-out objects and a novel human. Bottom: Breakdown of the failure modes of our approach.

Fig. 8. Human and robot demonstrations used for meta-training for the experiments in Section VI-B with large domain shift. We used ten different diverse backgrounds for collecting human demonstrations.

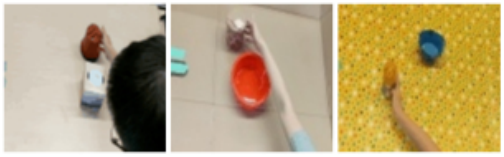

Fig. 9. Frames from the human demos used for evaluation in Section VI-B, illustrating the background scenes. The leftmost background was in the meta-training set (seen bg), whereas the right two backgrounds are novel (novel bg1 and novel bg2). The objects and human demonstrator are novel.

C. Sawyer Experiments

The goal of this experiment is to evaluate the generality of our method on a different robot and a different form of robot demonstration collection. We will use a 7-DoF Sawyer robot (see Figure 10), and use kinesthetic teaching to record the robot demonstrations, which introduces additional challenges due to the presence of the human demonstrator in the recorded images. The human demonstrations are collected from the perspective of the robot. We consider the placing task described in Section VI-A. Unlike the PR2 experiments, the action space will be a single commanded pose of the end-effector and we will use mean-squared error for the outer meta-objective. Since we have thoroughly compared our method on the PR2 benchmarks, we only evaluate our proposed method in this experiment. We evaluated our method using 18 held-out objects and 3 trials per object. The result was a 77.8% placing success rate, indicating that our method can successfully be applied to the Sawyer robot and can handle kinesthetic demonstrations during meta-training.

D. Learned Adaptation Objective Ablation

Finally, we study the importance of our proposed temporal adaptation objective. In order to isolate just the temporal adaptation loss, we perform this experiment in simulation, in a setting without domain shift. We use the simulated pushing task proposed by Finn et al. [15] in the MuJoCo physics engine [52]. To briefly summarize the experimental set-up, the imitation problem involves controlling a 7-DoF robot arm via torque-control to push one object to a fixed target position amid one distractor, using RGB images as input. The initial positions of the objects are randomized, as is the texture, shape, size, mass, and friction. Meta-training uses 105 object meshes, while 11 held-out meshes and multiple held-out textures are used for meta-testing. A push is considered successful if the target object lands on the target position for at least 10 timesteps within the 100-timestep episode. The results in Table III demonstrate a 14% absolute improvement in success by using a temporal adaptation objective, indicating the importance of integrating temporal information when learning from raw video.

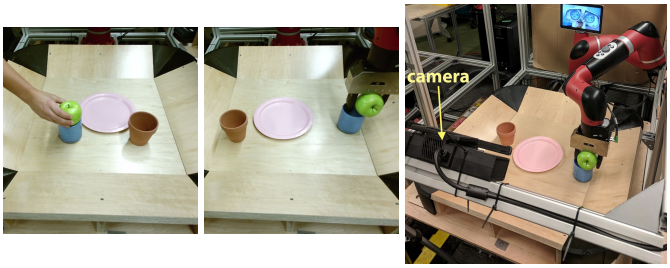

Fig. 10. Sawyer robot set-up. From left the right: a human demo from the robot's perspective, the policy execution from the robot's perspective, and an photo illustrating the experimental set-up.

	simulated pushing no domain shift
LSTM [10]	34.23%
contextual	56.98%
MIL, linear loss [15]	66.44%
MIL, temporal loss (ours)	**80.63%**

TABLE III. One-shot success rate of simulated 7-DoF pushing using video demonstrations with no domain shift

VII. Discussion

We presented an approach that enables a robot learning to visually recognize and manipulate a new object after observing just one video demonstration from a human user. To enable this, our method uses a meta-training phase where it acquires a rich prior over human imitation, using both human and robot demonstrations involving other objects. Our method extends a prior meta-learning approach to allow for learning cross-domain correspondences and includes a temporal adaptation loss function. Our experiments demonstrate that, after meta-learning, robots can acquire vision-based manipulation skills for a new object using from video of a human demonstrator in a substantially different setting.

Limitations: While our work enables one-shot learning for manipulating new objects from one video of a human, our current experiments do not yet demonstrate the ability to learn entirely new motions in one shot. The behaviors at meta-test time are structurally similar to those observed at meta-training time, though they may involve previously unseen objects and demonstrators. We expect that more data and a higher-capacity model would likely help enable such an extension. However, we leave this to future work. An additional challenge with our approach is the amount of demonstration data that is needed for meta-training. In our experiments, we used a few thousand demonstrations from robots and humans. However, the total amount of data *per-object* is quite low (around 10 trials), which is one or two orders of magnitude less than the number of demonstrations per-object used in recent single-task imitation learning works [36, 62]. Thus, if the goal is to enable a *generalist* robot that can adapt to a diverse range of objects, then our approach is substantially more practical.

Beyond Human Imitation: While our experiments focus on imitating humans, the proposed method is not specific to perceiving humans, and could also be used, for example, for imitating animals or a simulated robot, for simulation to real world transfer. Beyond imitation, we believe our approach is likely more broadly applicable to problems that involve inferring information from out-of-domain data, such as one-shot object recognition from product images, a problem faced by teams in the Amazon Robotics Challenge [61].

ACKNOWLEDGMENTS

We thank Saurabh Gupta, Rowan McAllister, and Dinesh Jayaraman for helpful feedback on an early version of this paper.

REFERENCES

[1] B. Akgun, M. Cakmak, J. W. Yoo, and A. L. Thomaz. Trajectories and keyframes for kinesthetic teaching: A human-robot interaction perspective. In *International Conference on Human-Robot Interaction*, 2012.

[2] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 2009.

[3] Y. Aytar and A. Zisserman. Tabula rasa: Model transfer for object category detection. In *International Conference on Computer Vision (ICCV)*, 2011.

[4] C. M. Bishop. Mixture density networks. 1994.

[5] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. *arXiv:1612.05424*, 2016.

[6] M. Brass and C. Heyes. Imitation: is cognitive neuroscience solving the correspondence problem? *Trends in cognitive sciences*, 2005.

[7] S. Calinon and A. Billard. Teaching a humanoid robot to recognize and reproduce social cues. In *International Symposium on Robot and Human Interactive Communication (ROMAN)*, 2006.

[8] S. Calinon, P. Evrard, E. Gribovskaya, A. Billard, and A. Kheddar. Learning collaborative manipulation tasks by demonstration using a haptic interface. In *International Conference on Advanced Robotics (ICAR)*, 2009.

[9] R. Dillmann. Teaching and learning of robot tasks via observation of human performance. *Robotics and Autonomous Systems*, 2004.

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

[11] S. Ekvall and D. Kragic. Interactive grasp learning based on human demonstration. In *International Conference on Robotics and Automation (ICRA)*, 2004.

[12] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *International Conference on Computer Vision (ICCV)*, 2013.

[13] C. Finn, X. Y. Tan, Y. Duan, T. Darrell, S. Levine, and P. Abbeel. Deep spatial autoencoders for visuomotor learning. In *International Conference on Robotics and Automation (ICRA)*, 2016.

[14] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *International Conference on Machine Learning (ICML)*, 2017.

[15] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine. One-shot visual imitation learning via meta-learning. *Conference on Robot Learning (CoRL)*, 2017.

[16] B. Gong, K. Grauman, and F. Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *International Conference on Machine Learning (ICML)*, 2013.

[17] E. Grant, C. Finn, S. Levine, T. Darrell, and T. Griffiths. Recasting gradient-based meta-learning as hierarchical bayes. In *International Conference on Learning Representations (ICLR)*, 2018.

[18] D.-K. Kim and M. R. Walter. Satellite image-based localization via learned embeddings. In *International Conference on Robotics and Automation (ICRA)*, 2017.

[19] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)*, 2015.

[20] H. Kjellstrom, J. Romero, and D. Kragic. Visual recognition of grasps for human-to-robot mapping. In *International Conference on Intelligent Robots and Systems*, 2008.

[21] V. Kruger, D. L. Herzog, S. Baby, A. Ude, and D. Kragic. Learning actions from observations. *IEEE Robotics & Automation Magazine*, 2010.

[22] J. Lee and M. S. Ryoo. Learning robot activities from first-person human videos using convolutional future regression. *arXiv:1703.01040*, 2017.

[23] K. Lee, Y. Su, T.-K. Kim, and Y. Demiris. A syntactic approach to robot imitation learning using probabilistic activity grammars. *Robotics and Autonomous Systems*, 2013.

[24] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end learning of deep visuomotor policies. *Journal of Machine Learning Research (JMLR)*, 2016.

[25] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales. Learning to generalize: Meta-learning for domain generalization. *arXiv:1710.03463*, 2017.

[26] Y. Liu, A. Gupta, P. Abbeel, and S. Levine. Imitation from observation: Learning to imitate behaviors from raw video via context translation. *arXiv:1707.03374*, 2017.

[27] K. Muelling, A. Venkatraman, J.-S. Valois, J. Downey, J. Weiss, S. Javdani, M. Hebert, A. B. Schwartz, J. L. Collinger, and J. A. Bagnell. Autonomy infused teleoperation with application to bci manipulation. *Autonomous Robots*, 2017.

[28] 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 (IJRR)*, 2013.

[29] C. L. Nehaniv, K. Dautenhahn, et al. The correspondence problem. *Imitation in animals and artifacts*, 2002.

[30] A. Nguyen, D. Kanoulas, L. Muratore, D. G. Caldwell, and N. G. Tsagarakis. Translating videos to commands for robotic manipulation with deep recurrent neural networks. *arXiv:1710.00290*, 2017.

[31] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal. Learning and generalization of motor skills by learning from demonstration. In *International Conference on Robotics and Automation (ICRA)*, 2009.

[32] P. Pastor, L. Righetti, M. Kalakrishnan, and S. Schaal. Online movement adaptation based on previous sensor experiences. In *International Conference on Intelligent Robots and Systems (IROS)*, 2011.

[33] V. M. Patel, R. Gopalan, R. Li, and R. Chellappa. Visual domain adaptation: A survey of recent advances. *IEEE signal processing magazine*, 2015.

[34] D. Pomerleau. ALVINN: an autonomous land vehicle in a neural network. In *Advances in Neural Information Processing Systems (NIPS)*, 1989.

[35] R. Rahmatizadeh, P. Abolghasemi, L. Bölöni, and S. Levine. Vision-based multi-task manipulation for inexpensive robots using end-to-end learning from demonstration. *arXiv:1707.02920*, 2017.

[36] R. Rahmatizadeh, P. Abolghasemi, A. Behal, and L. Bölöni. Learning real manipulation tasks from virtual demonstrations using lstm. *AAAI*, 2018.

[37] K. Ramirez-Amaro, M. Beetz, and G. Cheng. Transferring skills to humanoid robots by extracting semantic representations from observations of human activities. *Artificial Intelligence*, 2015.

[38] M. A. Rana, M. Mukadam, S. R. Ahmadzadeh, S. Chernova, and B. Boots. Towards robust skill generalization: Unifying learning from demonstration and motion planning. In *Proceedings of the 2017 Conference on Robot Learning (CoRL)*, 2017.

[39] N. Rhinehart and K. M. Kitani. First-person activity forecasting with online inverse reinforcement learning. *International Conference on Computer Vision*, 2017.

[40] J. Rothfuss, F. Ferreira, E. Erdal Askoy, Y. Zhou, and T. Asfour. Deep episodic memory: Encoding, recalling, and predicting episodic experiences for robot action execution. *arXiv:1801.04134*, 2018.

[41] F. Sadeghi and S. Levine. (cad)ˆ2 rl: Real single-image flight without a single real image. *Robotics: Science and Systems (R:SS)*, 2016.

[42] R. Salakhutdinov, J. Tenenbaum, and A. Torralba. One-shot learning with a hierarchical nonparametric bayesian model. In *ICML Workshop on Unsupervised and Transfer Learning*, 2012.

[43] R. J. Santos. Equivalence of regularization and truncated iteration for general ill-posed problems. *Linear algebra and its applications*, 1996.

[44] S. Schaal, A. Ijspeert, and A. Billard. Computational approaches to motor learning by imitation. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 2003.

[45] J. Schmidhuber. Evolutionary principles in self-referential learning. *Diploma thesis, Institut f. Informatik, Tech. Univ. Munich*, 1987.

[46] P. Sermanet, C. Lynch, J. Hsu, and S. Levine. Time-contrastive networks: Self-supervised learning from multi-view observation. *arXiv:1704.06888*, 2017.

[47] P. Sermanet, K. Xu, and S. Levine. Unsupervised perceptual rewards for imitation learning. *Robotics: Science and Systems (R:SS)*, 2017.

[48] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. Learning from simulated and unsupervised images through adversarial training. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[49] B. Stadie, P. Abbeel, and I. Sutskever. Third-person imitation learning. *International Conference on Learning Representations (ICLR)*, 2017.

[50] L. Tai, J. Zhang, M. Liu, and W. Burgard. Socially-compliant navigation through raw depth inputs with generative adversarial imitation learning. *arXiv:1710.02543*, 2017.

[51] S. Thrun and L. Pratt. *Learning to learn*. Springer Science & Business Media, 1998.

[52] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[53] A. Tow, N. Sünderhauf, S. Shirazi, M. Milford, and J. Leitner. What would you do? acting by learning to predict. *arXiv:1703.02658*, 2017.

[54] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv:1412.3474*, 2014.

[55] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv:1609.03499*, 2016.

[56] 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 (CoRL)*, 2017.

[57] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv:1711.00199*, 2017.

[58] Y. Yang, Y. Li, C. Fermüller, and Y. Aloimonos. Robot learning manipulation action plans by" watching" unconstrained videos from the world wide web. In *AAAI*, 2015.

[59] D. Yoo, N. Kim, S. Park, A. S. Paek, and I. S. Kweon. Pixel-level domain transfer. In *European Conference on Computer Vision*. Springer, 2016.

[60] Y. You, X. Pan, Z. Wang, and C. Lu. Virtual to real reinforcement learning for autonomous driving. *arXiv:1704.03952*, 2017.

[61] A. Zeng, S. Song, K.-T. Yu, E. Donlon, F. R. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo, et al. Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching. In *International Conference on Robotics and Automation (ICRA)*, 2018.

[62] T. Zhang, Z. McCarthy, O. Jow, D. Lee, K. Goldberg, and P. Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. *International Conference on Robotics and Automation (ICRA)*, 2018.

APPENDIX

In this appendix, we include experimental details and hyperparameters for each of the experiments.

All of the human and robot demonstrations are collecting at 10 Hz, and take approximately 3-8 seconds. Then, for meta-training, we randomly sampled 40 of the frames and corresponding actions to be used as the demonstration. The robot policy is executed at 10 Hz.

A. Placing, Pushing, and Pick-and-Place Experiments

We include the details of the experiments in Section VI-A for the placing, pushing, and pick-and-place tasks. The architecture and hyperparameters were selected by evaluating the end-effector loss and control loss on a validation set of 12, 8, 12 objects for placing, pushing, and pick-and-place respectively, sampled and held out from the training data. For placing and pushing, the inputs are RGB images with size 100×90, and for pick-and-place, we use RGB-D images with size 110×90. We train separate models for each of the three settings.

For placing, the policy architecture uses 5 convolutional layers with 64 3×3 convolutional filters in each layer where the first three layers are with stride 2 and the last two layer is with stride 1. The first convolutional layer uses pretrained weights from VGG-19. It also uses 3 fully connected layers of size 100, and a learned adaptation objective with two layers of 32 10×1 convolutional filters followed by one layer of 1×1 convolutions. For pushing, the policy architecture, uses the same convolutional network, 4 fully connected layers of size 50, and a learned adaptation objective with two layers of 10 and 20 10×1 convolutional filters for action and final gripper pose prediction respectively followed by one layer of 1×1 convolutions. For pick-and-place, the policy architecture uses the same convolutional network except that the first convolutional layer is not pretrained. It also uses 16 3×3 convolutional filters that operate on the depth input and concatenates the depth stream to the RGB stream after the first convolutional layer, channel-wise, following the approach by Zhang et al. [62]. For pick-and-place, we use two gripper poses – one intermediate, when the gripper contacts the item to pick, and one final pose at the end of the trajectory. The architecture also uses 4 fully connected layers of size 50, and a learned adaptation objective with two layers of 10, 30, and 30, 10×1 convolutional filters for action, final gripper pose and pickup gripper pose prediction respectively followed by one layer of 1×1 convolutions. All architectures use ReLU nonlinearities and layer normalization.

All the baseline methods use the same architecture for convolutional layers as DAML for each experiment. DAML with a linear adaptation objective also uses the same fully-connected architecture as DAML with the temporal adaptation objective except that its learned adaptation objective consists of one linear layer. The LSTM uses 512 LSTM hidden units for all experiments. The contextual model uses 3 fully connected layers with size 100 for all experiments.

For placing, we use a behavioral cloning loss as a combination of ℓ_1 and ℓ_2 losses, where the ℓ_2 loss is scaled down by a factor of 100, following prior work [15]. For the pushing and pick-and-place experiments, all methods use a mixture density network as mentioned in Section V after the last fully-connected layer with 20 modes and the negative log likelihood of the mixture density network as the behavioral cloning loss. At test time, at each time step, we sample 100 actions from the learned mixture distribution and choose the action with highest probability. For DAML with both a linear and temporal adaptation objective, we use a step size $\alpha = 0.01$ for placing and $\alpha = 0.005$ for pushing and pick-and-place with inner gradient clipping within the range $[-30, 30]$. We use $12, 10$, and 4 tasks in the meta batch at each iteration for placing, pushing and pick-and-place respectively. For all methods, we use 1 human demonstration and 1 robot demonstration for each sampled task. We train the model for 50k iterations for placing and placing, and 75k iterations for pick-and-place. We use 5 inner gradient update steps and a bias transformation with dimension 20 for all experiments. Since we don't have the robot state $\mathbf{s}$ for human demonstrations, we set the state input to be 0 when computing inner gradient update and feed the robot states into the policy when we update the policy parameters with robot demonstrations.

B. Diverse Human Demonstration Experiments

We include the details of the experiments in Section VI-B using diverse human demonstrations. For meta-training, eight pushing demonstrations are taken for each of 80 total objects grouped into 40 pairs. Each demo is shot in front of a randomly selected background, among 10 backgrounds. The viewpoint for the human demos is held fixed with a phone camera mounted on a tripod. Before being fed into the model during training images are modified with noise sampled uniformly from the range $[-0.3, 0.3]$ to their lighting. This color augmentation process helps the model perform more robustly in different light conditions. We use the same input image size and policy architecture as the pushing experiment described in Section VI-A and Appendix A.

C. Sawyer Robot Experiments

We include the details of the experiments in Section VI-C on the Sawyer robot. The primary differences between this experiment and previous placing experiments are the robot used and how demonstrations were collected. While PR2 robot demonstrations were taken using a teleoperation interface, the Sawyer arm was controlled kinesthetically by humans: the demonstrator guided the Sawyer arm to perform the goal action. Demonstrations were collected at 10 Hz. Saved at each timestep are a monocular RGB image taken by a Kinect sensor, the robot's joint angles, its joint velocities, and its gripper pose.

The architecture and hyperparameters were tuned by evaluating the gripper pose loss on a held out validation set of 20 objects. The policy architecture takes in RGB images of size 100×100. It uses the same convolutional and fully-connected layers as well as squared error behavioral cloning

loss as in the model for the PR2 placing experiment, and a learned adaptation objective with three layers of 32 20×1 convolutional filters followed by one layer of 1×1 convolutions. We use a step size $\alpha = 0.005$ with inner gradient clipping within the range $[-30, 30]$, 8 as the meta batch size, and 1 human demonstration as well as 1 robot demonstration for each sampled task. We train the model for 60k iterations.

During training, we augmented the images using random color augmentation, by adding noise uniformly sampled in $[-0.3, 0.3]$ to their hue, saturation, and value. The images used during evaluation were not modified. To control the robot during evaluation, the first image frame is used to predict the final end-effector pose of the robot. After the robot reaches the predicted gripper pose, the robot is controlled using the prediction actions, which are continuous end-effector velocities. At this point the gripper is opened and the robot drops the held item (hopefully) into the target container.

D. Simulated Pushing Experiment

Here, we include details on the experiments in Section VI-D. The pushing environment was introduced and open-sourced by Finn et al. [15]. The expert policy for collecting demonstrations was computed using reinforcement learning. Following [15], we compute the reported success rates over 74 tasks with 6 trials per task, totalling to 444 trials. The time horizon is $T = 100$. A trial is considered successful if the target object lands on the target position for at least 10 timesteps within the 100-timestep episode.

The inputs are RGB images of size 125×125. The policy architecture uses 3 convolutional layers with 16 5×5 filters and stride 2 followed by 1 convolutional layer with 32 5×5 filters with stride 1. It also has 2 fully-connected layers of size 400, and a learned adaptation objective with two layers of 64 10×1 convolutional filters followed by one layer of 1×1 convolutions. The policy operates on a 125×125 RGB image, along with the robot joint angles, joint velocities, and end-effector pose. The behavioral cloning loss is the mean squared error between the predicted actions and the ground truth robot commands. We use step size $\alpha = 0.01$ with inner gradient clipping within the range $[-20, 20]$, and one inner gradient update step. We use 15 as the meta batch size, and 2 different robot demonstrations for each sampled task. We train our policy for 30k iterations.